TRANSFORMERS PROVABLY LEARN TO INTERNALIZE CHAIN-OF-THOUGHT

Anonymous authors
Paper under double-blind review

ABSTRACT

Explicit chain-of-thought (CoT) reasoning improves language models' performance, but generating intermediate tokens increases inference cost. Implicit CoT (ICoT) progressively removes those tokens during training, aiming to internalize reasoning without explicitly generating the intermediate steps at inference. We ask whether a transformer can still learn efficiently as intermediate reasoning steps are progressively removed during training. We introduce Log-ICoT, which uses a given hierarchical decomposition of the task into simpler subproblems. The curriculum removes intermediate steps in geometric chunks aligned with the levels of this hierarchy. We establish sufficient conditions under which a simplified L-layer transformer trained with Log-ICoT learns k-parity, where $k = 2^L$, using a number of samples polynomial in the input length n and $L = \log_2 k$ training stages. The resulting model predicts in one forward pass rather than generating a linear number of reasoning tokens sequentially. Experiments illustrate layer-wise internalization on parity and demonstrate the curriculum on balanced ListOps and four-digit multiplication with pretrained GPT-2. Log-ICoT attains high answer accuracy with fewer training updates than vanilla ICoT and lower inference latency than explicit CoT.

1 INTRODUCTION

Chain-of-thought (CoT) reasoning improves language models' performance by allowing them to generate intermediate steps before a final answer (Wei et al., 2022). Reasoning models such as OpenAI o1 and DeepSeek-R1 further demonstrate the effectiveness of extended reasoning traces and additional inference-time computation (OpenAI, 2024; DeepSeek-AI, 2025). However, generating long traces token by token increases inference latency and computational cost (Deng et al., 2024). This motivates methods that retain useful intermediate computation without expressing every step in language.

Implicit CoT (ICoT) progressively removes reasoning tokens during fine-tuning. We refer to its standard curriculum, which removes reasoning tokens at a fixed rate, as *token-wise ICoT*. It enables models to improve on direct-answer training without generating a reasoning trace at inference (Deng et al., 2024). This empirical success raises a central theoretical question. *Can a transformer still learn efficiently as intermediate reasoning steps are progressively removed during training?* Token-wise ICoT also requires a number of stages linear in the reasoning-trace length. *Can a structured curriculum retain this learning efficiency with substantially fewer stages?*

Parity learning as a testbed. To examine these questions, we study k-parity learning. The target is the product of an unknown fixed subset of k among n input bits in $\{\pm 1\}$, with k proportional to n. With only final labels, polynomially many samples and finite-precision gradient-oracle queries cannot guarantee learning random k-parity (Shalev-Shwartz et al., 2017; Wies et al., 2022; Kim & Suzuki, 2024). The finite-sample result is restated in Proposition 1 of Appendix D. By contrast, explicit CoT supervision enables one-layer transformers to learn parity with polynomially many samples (Kim & Suzuki, 2024; Wen et al., 2024). This contrast makes parity a natural setting for asking whether internalization preserves polynomial sample complexity. Parity's CoT trace organizes pairwise intermediate computations into a hierarchy. This structure provides a natural setting for comparing level-wise removal with fixed-rate token removal.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

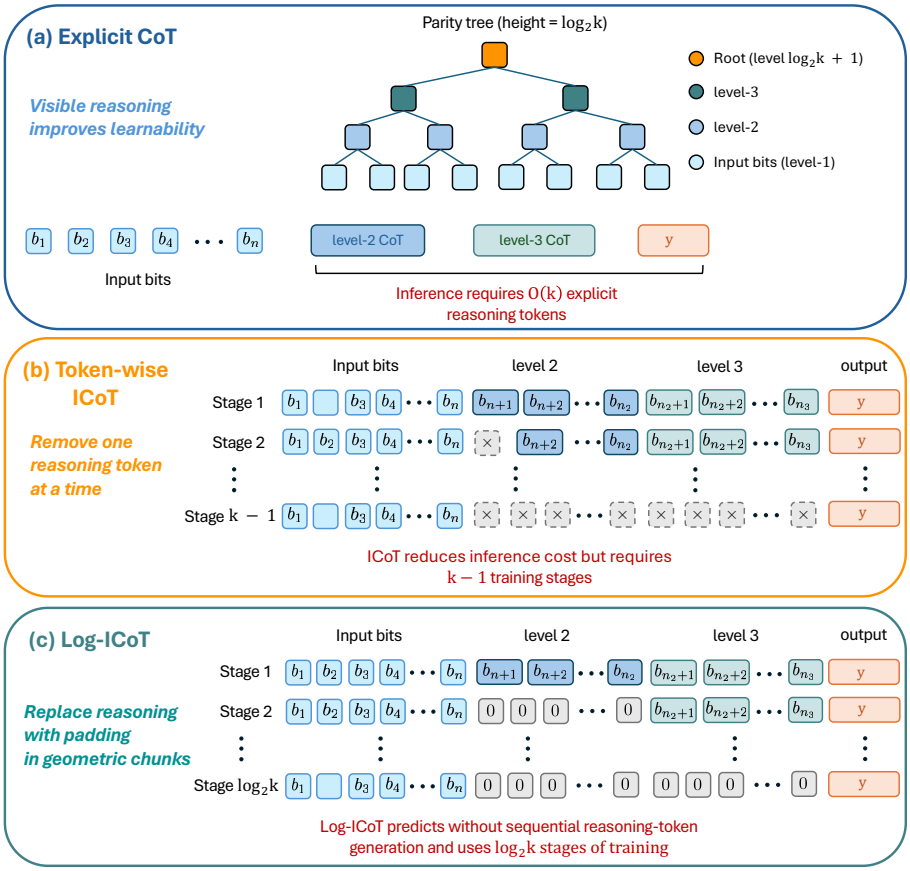

Figure 1: **Comparison of training paradigms on the k-parity task. (a)** Explicit CoT supervises the internal nodes of the parity tree, achieving sample-efficient learning at the cost of $\Omega(k)$ sequential reasoning tokens at inference. **(b)** Token-wise ICoT (Deng et al., 2024) removes intermediate tokens at a fixed rate. The one-token-per-transition example shown here uses $k-1$ training stages. Crosses mark removed tokens. **(c)** Log-ICoT replaces intermediate tokens with zero-valued padding in geometric chunks aligned with the tree levels, keeping the sequence length fixed. It uses $\log_2 k$ training stages and predicts without sequential reasoning-token generation.

Our contributions. We introduce Log-ICoT and establish a learning guarantee for internalizing hierarchical parity computations.

- **Log-ICoT curriculum.** We introduce a curriculum that removes intermediate reasoning steps in geometric chunks aligned with the parity tree's levels (Figure 1). It uses $L = \log_2 k$ training stages, whereas token-wise ICoT requires a number of stages linear in k (Deng et al., 2024).
- **Convergence guarantee for multi-layer ICoT.** We establish sufficient conditions under which a simplified L-layer transformer learns k-parity, with $k = 2^L$, from polynomially many samples in n (Theorem 1). This retains the polynomial sample complexity of explicit CoT (Kim & Suzuki, 2024; Wen et al., 2024). Our analysis explains how successive computation levels can be learned while preserving previously learned intermediate representations. The resulting model predicts in one forward pass through L layers instead of sequentially generating a number of reasoning tokens linear in k.
- **Empirical evidence.** Experiments illustrate layer-wise internalization on parity and demonstrate Log-ICoT beyond the theoretical setting on balanced ListOps (Nangia & Bowman, 2018) and four-digit multiplication with pretrained GPT-2. The curriculum attains high answer accuracy with fewer training updates than token-wise ICoT and lower inference latency than explicit CoT (Section 4).

1.1 Related Work

Learning parity and internalizing CoT. Wen et al. (2024); Kim & Suzuki (2024) show that explicit CoT supervision enables sample-efficient learning of parity with secret indices in one-layer transformers. Following the parity setting of Kim & Suzuki (2024), we study whether this sample efficiency is retained as intermediate computations are progressively internalized in multi-layer models.

Our curriculum builds on the progressive internalization method of Deng et al. (2024). Token-wise ICoT removes intermediate tokens at a fixed rate, whereas Log-ICoT uses geometric chunks aligned with the levels of a computation tree. Li et al. (2026) also study progressive internalization in an ICoT setup. Their negative result shows that internalization degrades the learning signal in a one-layer model. In contrast, our multi-layer construction retains sample-efficient learning by assigning one computation-tree level to each layer. The analysis also builds on Wang et al. (2025), who establish efficient learning of k-fold composition in multi-layer transformers using data of varying difficulty. Their local computation selects a single target token, whereas our parity nodes require balanced attention to two children.

CoT and latent reasoning. CoT can be elicited through prompting (Wei et al., 2022; Khot et al., 2022; Zhou et al., 2022) or learned from reasoning traces included in training data (Yue et al., 2023; Yu et al., 2023; Wang et al., 2023a; Shao et al., 2024). Related methods change how intermediate computation is represented. Pause and filler tokens provide additional computation positions (Goyal et al., 2023; Pfau et al., 2024), while planning tokens structure explicit reasoning steps (Wang et al., 2023b). Other approaches place intermediate reasoning in hidden states (Deng et al., 2023; 2024; Hao et al., 2024) or combine latent and textual steps (Su et al., 2025). Theoretical studies examine the expressivity of CoT and auxiliary tokens (Liu et al., 2022; Feng et al., 2023; Merrill & Sabharwal, 2023; Li et al., 2024b; London & Kanade, 2025) and CoT learning dynamics (Zhu et al., 2024b). Analyses of continuous CoT similarly study its reasoning capacity (Zhu et al., 2025b; Gozeten et al., 2025) and training dynamics (Zhu et al., 2025a). We study the learnability of progressively internalizing a discrete reasoning trace.

Transformer training dynamics. Training-dynamics analyses characterize how model parameters, attention, and representations evolve during optimization (Boix-Adsera et al., 2023; Tian et al., 2023; Guo et al., 2024). Studies of single attention layers characterize their learnable functions and mechanisms for next-token prediction and autoregressive generation (Fu et al., 2023; Li et al., 2024a; Ildiz et al., 2024). Task-specific analyses study the acquisition of associative memory (Bietti et al., 2023) and provide learning guarantees for sparse token selection and contextual bigrams (Wang et al., 2024; Ren et al., 2024), as well as in-context regression (Zhang et al., 2024). Reasoning-focused work examines two-hop inference in context (Guo et al., 2025) and CoT-based multi-step gradient descent for in-context regression (Huang et al., 2025a). Further analyses connect factual generalization to parameterization and training data (Zhu et al., 2024b; Huang et al., 2025b; Ma et al., 2026).

2 Preliminaries

Notations. For any integer $N > 0$, $[N] := \{1, 2, \ldots, N\}$. We use lower-case and upper-case bold letters (e.g., $\boldsymbol{a}$, $\boldsymbol{A}$) for vectors and matrices. Let $\boldsymbol{e}_{d,i} \in \mathbb{R}^d$ denote the standard basis vector with a 1 in the i-th coordinate and $\boldsymbol{I}_K \in \mathbb{R}^{K\times K}$ the identity. The multi-linear inner product of vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r \in \mathbb{R}^d$ is defined as

$$\langle \boldsymbol{x}_1, \ldots, \boldsymbol{x}_r \rangle := \sum_{i=1}^{d} \prod_{j=1}^{r} x_{j,i},$$

which generalizes the standard inner product: $\langle \boldsymbol{x}_1 \rangle = \boldsymbol{x}_1^\top \mathbf{1}_d$ and $\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle = \boldsymbol{x}_1^\top \boldsymbol{x}_2$.

2.1 Task Settings

Assume $n \geq k \geq 2$ with $k = \Theta(n)$, and let $\mathcal{S}$ denote the set of all size-k subsets of $[n]$. Following Kim & Suzuki (2024); Wen et al. (2024), we consider the k-parity problem: fix an unknown index set

$S \in \mathcal{S}$ and draw inputs $\boldsymbol{b} \sim \mathsf{Unif}(\{\pm 1\}^n)$ with labels $y = p_S(\boldsymbol{b}) := \prod_{j \in S} b_j$. The goal is to predict p_S on fresh inputs. Here n is the input length and $k = |S|$ controls task hardness. For our model, we take $k = 2^L$ and reserve $T = n + k - 1$ sequence positions, with B denoting the batch size. The intermediate positions are specified by the parity-tree construction in Section 3.1. For any predictor f_θ, we evaluate the population L_2 error

$$\mathcal{E}_S(\theta) := \mathbb{E}_{\boldsymbol{b}}\big[(p_S(\boldsymbol{b}) - f_\theta(\boldsymbol{b}))^2\big].$$

2.2 Model

Architecture. We consider an L-layer simplified transformer following Wang et al. (2025). To process the entire dataset in parallel, we stack the j-th bit across all B samples into a vector $\boldsymbol{b}_j = (b_j^1, \ldots, b_j^B)^\top \in \mathbb{R}^B$, so the input data $\boldsymbol{D} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_T] \in \mathbb{R}^{B \times T}$ is processed as a single batch using a shared positional encoding.

Embedding layer. For $T = n + k - 1$, set the embedding dimension $d := T + B(L+1)$. For each $j \in [T]$, the embedding is

$$\boldsymbol{x}_j = \boldsymbol{E}(\boldsymbol{D})_j = \begin{bmatrix} \boldsymbol{p}_j \\ \boldsymbol{b}_j \\ \boldsymbol{0}_{B \cdot L} \end{bmatrix} \in \mathbb{R}^d,$$

where $\boldsymbol{p}_j := \boldsymbol{e}_{T,j} \in \mathbb{R}^T$ is a one-hot positional encoding and the final $B \cdot L$ dimensions are reserved for the residual stream (Elhage et al., 2021).

Attention layer. We use a single-head attention per layer, reparameterizing the key-query matrix as $\boldsymbol{W}_{\mathsf{KQ}} := \boldsymbol{W}_{\mathsf{K}} \boldsymbol{W}_{\mathsf{Q}}^\top$ following Tian et al. (2023); Zhu et al. (2024b); Li et al. (2024a). We optimize only the upper-left $T \times T$ block $\boldsymbol{W} \in \mathbb{R}^{T \times T}$, fixing all other entries to zero so that attention scores depend only on positional encodings. The value matrix is the identity:

$$\boldsymbol{W}_{\mathsf{KQ}} = \begin{bmatrix} \boldsymbol{W} & \boldsymbol{0}_{T \times B_L} \\ \boldsymbol{0}_{B_L \times T} & \boldsymbol{0}_{B_L \times B_L} \end{bmatrix}, \quad \boldsymbol{W}_{\mathsf{V}} = \boldsymbol{I}_d, \quad B_L := B(L+1).$$

Definition 1 (Causal Self-Attention)**.** *Given $\boldsymbol{W} \in \mathbb{R}^{T \times T}$ and $\boldsymbol{X} \in \mathbb{R}^{d \times T}$, define causal self-attention by*

$$\mathrm{Attn}(\boldsymbol{X}; \boldsymbol{W}) = \boldsymbol{X} \cdot \mathbb{S}(\boldsymbol{X}^\top \boldsymbol{W}_{\mathsf{KQ}} \boldsymbol{X} + C),$$

where $\mathbb{S}(\cdot)$ is the column-wise softmax and $C \in \mathbb{R}^{T \times T}$ is the standard causal mask[1]

$$C_{j,m} = \begin{cases} 0 & \text{if } j < m, \\ -\infty & \text{otherwise.} \end{cases}$$

MLP layer. The MLP layer applies a fixed link function that maps the average of two signs to their product.

Definition 2 (Link function)**.** *Let $r := \tanh(1/2)$. We use the smooth, even degree-eight polynomial $\phi : [-1, 1] \to [-1, 1]$ satisfying*

$$\phi(\pm 1) = 1, \qquad \phi(0) = \phi(\pm r) = -1, \tag{1}$$

$$\phi'(z) = 0, \qquad z \in \mathcal{Z} := \{-1, -r, 0, r, 1\}. \tag{2}$$

Around zero, $\phi(x) = -1 + cx^2 + O(x^4)$ and $\phi'(x) = 2cx + O(|x|^3)$ for a constant $c > 0$. The explicit formula is given in Appendix B.2.

[1] At position $m = 1$, the attention sum has no keys. We set its output to the zero vector. The gate at this input position is zero, so the convention does not affect the model output.

Gated connections. We use a position-wise gated connection (Srivastava et al., 2015; Parisotto et al., 2020). Each layer ℓ has a gate vector $\boldsymbol{g}^{(\ell)} \in \mathbb{R}^T$ that mixes the block update $\boldsymbol{A}$ with the existing representation $\boldsymbol{B}$ at each position:

$$\mathsf{GC}(\boldsymbol{A}, \boldsymbol{B}; \boldsymbol{g}) := \boldsymbol{A}\mathrm{diag}(\boldsymbol{g}) + \boldsymbol{B}(\boldsymbol{I}_T - \mathrm{diag}(\boldsymbol{g})).$$

Definition 3 (Multi-layer Transformer)**.** *Let L be the depth. The trainable parameters are $\boldsymbol{\theta} := \{\boldsymbol{W}^{(\ell)}\}_{\ell \in [L]}$. Initializing $\boldsymbol{X}^{(0)} = \boldsymbol{E}(\boldsymbol{D}) \in \mathbb{R}^{d \times T}$, the layer update is*

$$\boldsymbol{X}^{(\ell)} = \boldsymbol{X}^{(\ell-1)} + \boldsymbol{W}_{\mathsf{O}}^{(\ell)} \mathsf{GC}\Big(\phi(\mathsf{Attn}(\boldsymbol{X}^{(\ell-1)}; \boldsymbol{W}^{(\ell)})), \boldsymbol{X}^{(\ell-1)}; \boldsymbol{g}^{(\ell)}\Big), \quad \ell \in [L].$$

The network output is $\mathcal{T}_{\boldsymbol{\theta}}(\boldsymbol{D}) = \boldsymbol{X}^{(L)}$. For $m \in [T]$, the output at position m is

$$\mathcal{T}_{\boldsymbol{\theta}}(\boldsymbol{D})_m = \boldsymbol{x}_m + \sum_{\ell=1}^{L} \boldsymbol{W}_{\mathsf{O}}^{(\ell)} \left(g_m^{(\ell)} \phi\Big(\sum_{j=1}^{m-1} \sigma_j(\boldsymbol{w}_m^{(\ell)}) \boldsymbol{x}_j^{(\ell-1)} \Big) + (1 - g_m^{(\ell)}) \boldsymbol{x}_m^{(\ell-1)} \right),$$

where $w_{j,m}^{(\ell)} = \boldsymbol{p}_j^\top \boldsymbol{W}^{(\ell)} \boldsymbol{p}_m$, and $\sigma_j(\boldsymbol{w}_m^{(\ell)})$ is the corresponding attention score. The readout layer $\Psi_\ell \in \mathbb{R}^{d \times B}$ decodes the output as $f_{\boldsymbol{\theta}}^{(\ell)}(\boldsymbol{D}) = \Psi_\ell^\top \mathcal{T}_{\boldsymbol{\theta}}(\boldsymbol{D})$.

Block-shift output and readout. We fix $\boldsymbol{W}_{\mathsf{O}}^{(\ell)}$ as a block-shift operator that moves content from residual block $[\ell]$ to block $[\ell+1]$:

$$\boldsymbol{W}_{\mathsf{O}}^{(\ell)} = \begin{bmatrix} \boldsymbol{0}_{T \times T} & \boldsymbol{0}_{T \times B_L} \\ \boldsymbol{0}_{B_L \times T} & \boldsymbol{e}_{L+1,\ell+1} \boldsymbol{e}_{L+1,\ell}^\top \otimes \boldsymbol{I}_B \end{bmatrix},$$

and the readout Ψ_ℓ as a selector for block $[\ell+1]$:

$$\Psi_\ell = \begin{bmatrix} \boldsymbol{0}_{T \times B} \\ \boldsymbol{e}_{L+1,\ell+1} \otimes \boldsymbol{I}_B \end{bmatrix}.$$

With this construction, $\Psi_{\ell'}^\top \boldsymbol{W}_{\mathsf{O}}^{(\ell)}$ is non-zero iff $\ell = \ell'$, so the readout at stage ℓ extracts exactly the content written by layer ℓ. By Lemma 10, gradients at well-trained layers are exponentially small, so we can focus on training one layer per stage.

3 Theoretical Results

3.1 Training Scheme

Task decomposition. We decompose the task into a hierarchy of two-parities (Figure 1). The task forms a complete binary tree of height L with $2k-1$ total nodes. Leaf nodes at level 1 represent the input bits $\{b_j : j \in S\}$, and the $k-1$ internal nodes are indexed $b_{n+1}, \ldots, b_T$ from bottom to top, left to right. We write $n_\ell := n + k(1 - 2^{-(\ell-1)})$ for the right endpoint of level ℓ, so $n_1 = n$ and $n_{L+1} = T$. For each position $j \in [T]$, let $p[j]$ be its parent in the tree if it exists, and set $p[j] = \emptyset$ otherwise. For each internal node b_m, let $c_1[m], c_2[m]$ be its children, with $c_1[m] < c_2[m] < m$. Set $h[j] = 1$ for $j \le n$, and for $m > n$ define $h[m]$ by $n_{h[m]-1} < m \le n_{h[m]}$. Each internal node satisfies $b_m = b_{c_1[m]} b_{c_2[m]}$, and the root b_T equals the label y. The levels range from 1 to $L+1$, with layer ℓ computing tree level $\ell+1$. The $n-k$ input positions outside S are not leaves. Figure 3 in Appendix B.1 shows an example.

Chain-of-Thought (CoT) for parity. The parity tree gives a natural sequence of intermediate computations. We expand the input vector $\boldsymbol{b} \in \{\pm 1\}^n$ to a length-T sequence $\boldsymbol{b} \in \{\pm 1\}^T$ by appending the internal node values $b_{n+1}, \ldots, b_T$, where each b_m $(m > n)$ is computed via $b_m = b_{c_1[m]} b_{c_2[m]}$. Each training sample is supplied with this CoT trace. CoT supervision trains the model to predict each b_m at position m (Kim & Suzuki, 2024; Wen et al., 2024).

Algorithm 1 Log-ICoT training

Require: Per-stage batch size B, learning rates $(\eta_t)_{t=1}^{L}$, approximate stage-gradient oracle $\tilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}^{(t)}$, entrywise nearest-integer rounding q

1: Initialize $\boldsymbol{\theta}(0) = \{\boldsymbol{W}^{(\ell)}(0) = 0_{T\times T}\}_{\ell=1}^{L}$
2: **for** $t = 1, \ldots, L$ **do**
3: Draw B fresh inputs with full CoT traces and form $\boldsymbol{D}$ by zero-padding positions $n+1, \ldots, n_t$
4: $\boldsymbol{\theta}(t) \leftarrow q\Big(\boldsymbol{\theta}(t-1) - \eta_t\, \tilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}^{(t)}(\boldsymbol{\theta}(t-1))\Big)$
5: **end for**
6: **return** $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}(L)$

Implicit Chain-of-Thought (ICoT). ICoT (Deng et al., 2024) progressively removes intermediate CoT steps during training, internalizing the reasoning into hidden states so that at inference the model produces the answer directly without explicit reasoning. In the original formulation, removed CoT steps shorten the sequence. We instead pad with zeros, keeping the length fixed at T. The model produces the answer in a single forward pass through L layers. This use of padding is motivated by Merrill & Sabharwal (2025), who show that padding positions can support parallel intermediate computation in a related attention model. Concretely, each parity-tree level is computed in parallel across positions with one level per transformer layer. This contrasts with autoregressive CoT generation, which would require $O(k)$ sequential steps.

Log-ICoT curriculum. We adopt the Log-ICoT curriculum in Figure 1(c), replacing CoT values with zero padding in *geometric increments* aligned with the parity tree's levels:

- Stage 1: Full CoT (all positions $n+1, \ldots, T$ shown).
- Stage t ($2 \le t \le L$): Replace positions $n+1, \ldots, n_t$ with padding (zeros), keeping positions $n_t+1, \ldots, T$ shown.

At $t = L$, only the root position T remains supervised. Since $L = \log_2 k$, the curriculum contains only a logarithmic number of stages, in contrast to the linear number used in token-wise ICoT (Deng et al., 2024).

Loss function. At stage t, positions $n+1, \ldots, n_t$ are padded, while the CoT values at positions $m > n_t$ remain in the training sequence. We call the latter the *supervised positions* and compare their readouts with their labels. The stage loss is

$$\mathcal{L}^{(t)}(\boldsymbol{\theta}) = \frac{1}{2B} \sum_{m=n_t+1}^{T} \|\Psi_t^{\top} \mathcal{T}_{\boldsymbol{\theta}}(\boldsymbol{D})_m - \boldsymbol{b}_m\|^2, \tag{3}$$

where $\boldsymbol{D}$ denotes the stage-t padded batch, $\boldsymbol{b}_m \in \mathbb{R}^B$ stacks the ground-truth parities at position m, and $\Psi_t^{\top} \mathcal{T}_{\boldsymbol{\theta}}(\boldsymbol{D})_m \in \mathbb{R}^B$ stacks the corresponding predictions.

Algorithm. We use approximate stage gradients $\tilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}^{(t)}$ satisfying $\|\tilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}^{(t)} - \nabla_{\boldsymbol{\theta}}\mathcal{L}^{(t)}\|_\infty \le \delta_n$ at each query, where the norm takes the maximum over trainable coordinates. At each stage t, we draw an independent fresh batch of size B and take one gradient step on $\mathcal{L}^{(t)}$, followed by integer quantization (Algorithm 1). Here $q : \mathbb{R} \to \mathbb{Z}$ rounds to the nearest integer and acts entrywise on the trainable weights. The curriculum has $L = \log_2 k$ stages. Thus B denotes the per-stage batch size. Since fresh samples are used at each stage, the total number of training samples used by the curriculum is $BL = O(B \log k)$.

Evaluation. At test time, we fill positions $1, \ldots, n$ with random input bits and zero-fill all $T - n = k - 1$ CoT input positions, including the root slot T. The prediction is read out at the root, whose label is a target rather than an input at inference. Concretely, given an independent test batch $\boldsymbol{D}_{\mathsf{test}} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n, \boldsymbol{0}, \ldots, \boldsymbol{0}] \in \mathbb{R}^{B\times T}$ with $\boldsymbol{b}_j \sim \mathsf{Unif}(\{\pm 1\}^B)$, the target is $(\boldsymbol{y}_{\mathsf{test}})_i = \prod_{j\in S} b_j^i$ for $i \in [B]$.

3.2 Key Challenges

Learning parity across multiple layers presents two obstacles. Zero-initialized attention logits make later input positions attend uniformly to preceding tokens. The resulting averages can obscure the position-specific information needed to identify each node's children. Later curriculum stages can also disturb previously trained layers, making the intermediate representations they supply unreliable.

Representation collapse. At sufficiently late input positions, the uniform average of preceding mean-zero bits concentrates near zero. Since $\phi(0) = -1$ and $\phi'(0) = 0$, the link satisfies $\phi(z) + 1 = O(z^2)$ near zero (Definition 2). The concentration argument in Appendix C.1 then places the linked average near $-\mathbf{1}_B$. Without a gate, the residual stream carries this near-constant block into later layers. It can then introduce an $O(n^{-1})$ noise term in their gradients, obscuring the $O(n^{-2})$ signal that identifies the two children. To prevent this, we set the position-wise gates introduced in Section 2.2 by tree level:

$$g_m^{(\ell)} = \begin{cases} 1, & n_\ell + 1 \le m \le n_{\ell+1}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Layer ℓ writes only at its assigned level. At other positions, the zero gate passes the previous representation through the block-shift map rather than writing the link output. Input bits thus remain available to later layers.

Error propagation. Following Kim & Suzuki (2024), we round the attention weights to integers after each gradient step (Algorithm 1). Within a stage, the chosen learning rate and gradient-error bounds make child-logit updates round to nonzero integers and distractor updates round to zero. Across stages, the link's derivative vanishes at ideal attention averages (Definition 2). The resulting bound in Lemma 10 keeps subsequent updates to previously trained weights below the rounding threshold, preserving their integer values and the intermediate representations they produce.

3.3 Main Theorem

We work in the regime $k = \Theta(n)$, so $L = \log_2 k = \Theta(\log n)$ and $T = n + k - 1 = \Theta(n)$. Let $g = 1 + \log_2 \|\phi'\|_\infty$. The final prediction is read out at the root position T. By Lemma 4, its gradient with respect to the trainable attention weights has norm $O(n^g)$, meeting the condition in Proposition 1. With only final labels, that proposition shows that polynomially many samples and finite-precision gradient-oracle queries cannot guarantee learning random k-parity. The following theorem instead gives a learning guarantee with intermediate supervision for every fixed S.

Theorem 1 (Log-ICoT). *Fix any $S \in \mathcal{S}$ with $k = 2^L$ and consider the L-layer transformer of Definition 3 with gates specified by (4). Suppose the per-stage batch size satisfies $B = \Omega(n^{2+\epsilon})$ for some constant $0 < \epsilon \le 1$, the stage-gradient oracle has coordinatewise error $\delta_n = O(n^{-2-\epsilon/8})$, and at each stage $t \in [L]$, the learning rate is set to $\eta_t = \frac{K_n n_t^2}{2c}$, where $K_n := \lceil n^{\epsilon/16} \rceil$ and $c > 0$ is the link function constant in Definition 2. Then for sufficiently large n, with probability $1 - \exp(-\Omega(n^{\epsilon/2}))$ over the random sampling of the training data, the output $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}(L)$ of Algorithm 1 satisfies*

$$\|f_{\hat{\boldsymbol{\theta}}}^{(L)}(\boldsymbol{D}_{\text{test}})_T - \boldsymbol{y}_{\text{test}}\|_\infty \;\le\; \exp(-\Omega(n^{\epsilon/16})).$$

4 Experiments

We evaluate Log-ICoT on parity with a structured transformer, and on balanced ListOps and four-digit multiplication with pretrained GPT-2. Unlike standard ListOps (Nangia & Bowman, 2018), our balanced variant fixes two children per operator and equal leaf depth. The parity experiment tests the layer-wise internalization suggested by our analysis in Section 3. The latter tasks test whether the curriculum extends beyond the theoretical setting while retaining answer accuracy, requiring fewer training updates than token-wise ICoT, and reducing inference latency relative to explicit CoT. Implementation details appear in Appendix B.

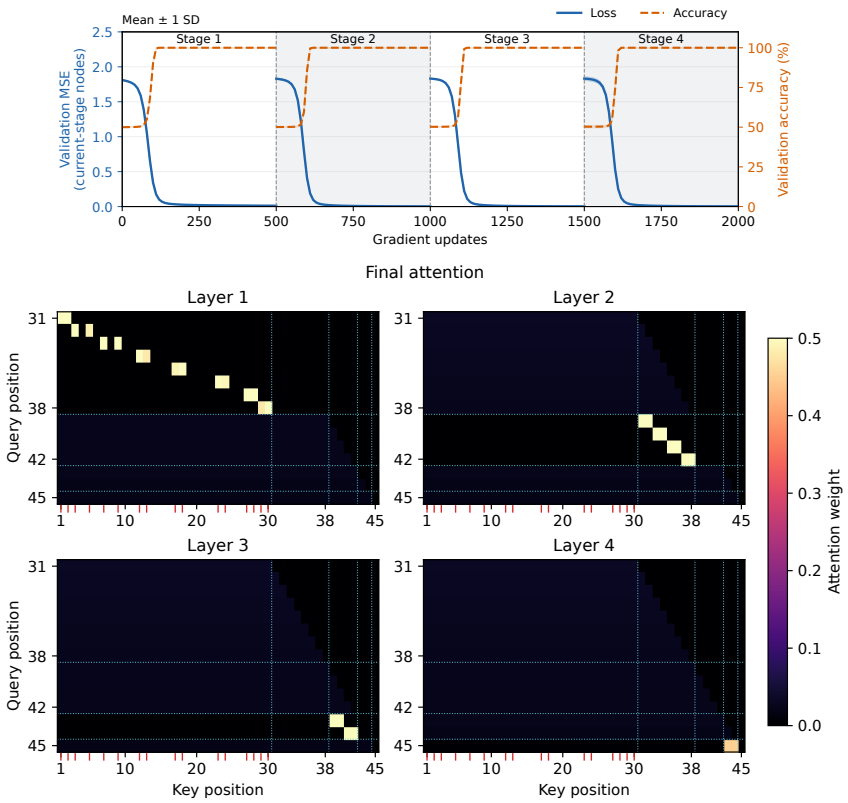

Figure 2: **Parity training dynamics and learned attention with the quartic link.** The top panel shows validation MSE per current-stage node (left axis) and validation accuracy (right axis) across four Log-ICoT stages. Curves show three-seed means, with bands indicating one sample standard deviation. Dashed lines mark stage boundaries. The bottom panels show final-stage softmax attention from one representative run in each layer, with rows indexing queries, columns indexing keys, dotted lines marking tree-level boundaries, and red ticks marking the secret set S, at $n = 30$, $k = 16$.

4.1 Layer-wise internalization on parity

The parity experiments use the model in Definition 3 with a simpler quartic link in place of the degree-eight link in Definition 2. The link and training settings appear in Appendix B.2.

As successive tree levels are padded, validation MSE recovers within each stage (Figure 2). The quartic-link runs reach 100% final root accuracy across three seeds (Appendix B.2). A representative final-stage attention map concentrates on the two children of each node at the corresponding tree level (Figure 2).

Removing the gates while retaining the quartic link leads to near-chance accuracy, consistent with the representation collapse in Section 3.2. Training the full key-query matrix also fails with the fixed link. In contrast, replacing the link with a learned MLP permits ungated internalization with either positional or full key-query training (Table 2).

4.2 Beyond parity

To assess whether layer-wise internalization extends to pretrained transformers, we consider tasks that retain a hierarchical decomposition but involve richer operations and intermediate values. We fine-tune pretrained GPT-2 small (Radford et al., 2019) on balanced ListOps (Nangia & Bowman, 2018) and four-digit multiplication, using supplied bottom-up reasoning traces. We evaluate ListOps at depths 4 and 5. Multiplication recursively combines single-digit products and partial sums into a reasoning tree, detailed in Appendix B.4.

Table 1: **Validation accuracy, training updates, and inference cost on GPT-2 small.** Accuracy and updates are three-seed means ± sample standard deviations. Updates are in thousands, counted up to the final saved checkpoint. Latency and dense FLOPs include prefill and cached decoding at batch size 1 on one H200, with speedups relative to explicit CoT. Data and training details appear in Appendix B.4.

Task	Method	Val. acc. (%)	Updates (10^3)	Latency (ms)	GFLOPs	Speedup
ListOps, $D = 4$	Explicit CoT	99.7 ± 0.1	3.5 ± 0.0	70.4	20.5	$1.0\times$
	Token-wise ICoT	99.8 ± 0.2	98.8 ± 0.0	10.9	16.2	$6.5\times$
	Log-ICoT	97.9 ± 0.3	29.8 ± 5.9	10.9	20.5	$6.5\times$
ListOps, $D = 5$	Explicit CoT	100.0 ± 0.0	3.5 ± 0.0	124.4	41.1	$1.0\times$
	Token-wise ICoT	96.3 ± 3.0	550.4 ± 0.0	10.7	32.5	$11.6\times$
	Log-ICoT	98.2 ± 0.2	50.9 ± 6.1	10.9	41.2	$11.4\times$
4×4 mult.	Explicit CoT	100.0 ± 0.0	3.0 ± 1.0	323.8	27.4	$1.0\times$
	Token-wise ICoT	99.6 ± 0.4	161.7 ± 7.6	26.5	5.0	$12.2\times$
	Log-ICoT	99.3 ± 0.5	53.3 ± 10.4	26.7	27.3	$12.1\times$

We compare Log-ICoT with explicit CoT and token-wise ICoT (Deng et al., 2024). All methods start with explicit-CoT training, which retains the reasoning trace. Token-wise ICoT removes tokens from the front. Log-ICoT removes tree levels, retaining hidden positions as padding and smoothing transitions with token-wise removal. Both ICoT methods generate only the answer at inference. We report greedy-decoding validation accuracy and cumulative gradient updates up to the final saved checkpoint, including initial explicit-CoT training and curriculum transitions.

On depth-4 ListOps, Log-ICoT reaches 97.9% mean validation accuracy, compared with 99.8% for token-wise ICoT, while using $3.3\times$ fewer updates (Table 1). On four-digit multiplication, it reaches 99.3% versus 99.6% with $3.0\times$ fewer updates. At depth 5, Log-ICoT reaches 98.2% versus 96.3% for token-wise ICoT with $10.8\times$ fewer updates. The token-wise depth-5 result varies across seeds, as detailed in Appendix B.4. Both internalization methods require more updates than explicit CoT, but avoid generating the reasoning trace at inference.

4.3 Inference cost

As introduced in Section 3.1, Log-ICoT retains internalized trace positions as padding and processes them during prompt prefill rather than generating them autoregressively.

On depth-4 ListOps and four-digit multiplication, Log-ICoT achieves accuracy comparable to explicit CoT with $6.5\times$ and $12.1\times$ lower batch-1 latency, respectively (Table 1). These measurements include both prompt prefill and answer decoding on one H200. Depth-5 ListOps yields an $11.4\times$ speedup. Log-ICoT's batch-1 latency is also comparable to token-wise ICoT.

The latency reduction comes from replacing sequential decoding with prefill rather than reducing total arithmetic work. Log-ICoT's dense FLOPs remain close to those of explicit CoT and exceed those of token-wise ICoT. The no-padding variant reduces sequence length but uses more training updates in the ListOps comparison (Appendix B.4).

5 Conclusion

We establish sufficient conditions for a simplified multi-layer transformer to internalize parity computations with polynomial sample complexity. Log-ICoT removes intermediate reasoning in geometric chunks aligned with a given task hierarchy. The construction uses a logarithmic number of curriculum stages and predicts the answer in one forward pass. Parity experiments illustrate layer-wise internalization, while experiments on balanced ListOps and four-digit multiplication extend the curriculum to pretrained GPT-2. On these tasks, Log-ICoT attains high answer accuracy with fewer training updates than token-wise ICoT and lower inference latency than explicit CoT. Extending the theory to learned decompositions and less constrained architectures remains open (Appendix A).

AI USE STATEMENT

In this work, we used generative AI tools for editing and polishing the writing, for assistance with writing experiment and analysis code, and for literature search. All AI-assisted text and code were reviewed by the authors, and every cited work was checked against its source. We have not used generative AI tools for research ideation, and the theoretical results and proofs were derived by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies reasoning internalization through theoretical analysis and experiments on synthetic tasks using publicly available model weights. Our experiments involve no human subjects and collect no personal data. We expect the direct ethical risks of these synthetic-task experiments to be limited, while recognizing that broader applications of improved reasoning methods may raise dual-use concerns.

REPRODUCIBILITY STATEMENT

The theoretical setup and training algorithm are specified in Section 2 and Section 3.1. Supporting proofs appear in Appendix C, and the final-label lower-bound setup is given in Appendix D. Experimental data, task decompositions, training schedules, stopping criteria, and inference measurement protocols are described in Appendix B. Training results are reported as means and sample standard deviations across three seeds. Code and configurations will be released.

REFERENCES

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *Advances in Neural Information Processing Systems*, 36: 1560–1588, 2023.

Enric Boix-Adsera, Etai Littwin, Emmanuel Abbe, Samy Bengio, and Joshua Susskind. Transformers learn through gradual rank increase. *Advances in Neural Information Processing Systems*, 36: 24519–24551, 2023.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36:70757–70798, 2023.

Hengyu Fu, Tianyu Guo, Yu Bai, and Song Mei. What can a single attention layer learn? a study through the random features lens. *Advances in Neural Information Processing Systems*, 36: 11912–11951, 2023.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. *arXiv preprint arXiv:2310.02226*, 2023.

Halil Alperen Gozeten, M Emrullah Ildiz, Xuechen Zhang, Hrayr Harutyunyan, Ankit Singh Rawat, and Samet Oymak. Continuous chain of thought enables parallel exploration and reasoning. *arXiv preprint arXiv:2505.23648*, 2025.

Tianyu Guo, Druv Pai, Yu Bai, Jiantao Jiao, Michael I Jordan, and Song Mei. Active-dormant attention heads: Mechanistically demystifying extreme-token phenomena in llms. *arXiv preprint arXiv:2410.13835*, 2024.

Tianyu Guo, Hanlin Zhu, Ruiqi Zhang, Jiantao Jiao, Song Mei, Michael I Jordan, and Stuart Russell. How do llms perform two-hop reasoning in context? *arXiv preprint arXiv:2502.13913*, 2025.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Jianhao Huang, Zixuan Wang, and Jason D Lee. Transformers learn to implement multi-step gradient descent with chain of thought. *arXiv preprint arXiv:2502.21212*, 2025a.

Yixiao Huang, Hanlin Zhu, Tianyu Guo, Jiantao Jiao, Somayeh Sojoudi, Michael I Jordan, Stuart Russell, and Song Mei. Generalization or hallucination? understanding out-of-context reasoning in transformers. *arXiv preprint arXiv:2506.10887*, 2025b.

M Emrullah Ildiz, Yixiao Huang, Yingcong Li, Ankit Singh Rawat, and Samet Oymak. From self-attention to markov models: Unveiling the dynamics of generative transformers. *arXiv preprint arXiv:2402.13512*, 2024.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*, 2022.

Juno Kim and Taiji Suzuki. Transformers provably solve parity efficiently with chain of thought. *arXiv preprint arXiv:2410.08633*, 2024.

Juncai Li, Ru Li, Yuxiang Zhou, Boxiang Ma, and Jeff Z. Pan. Chain of thought compression: A theoretical analysis. *arXiv preprint arXiv:2601.21576*, 2026.

Yingcong Li, Yixiao Huang, Muhammed E Ildiz, Ankit Singh Rawat, and Samet Oymak. Mechanics of next token prediction with self-attention. In *International Conference on Artificial Intelligence and Statistics*, pp. 685–693. PMLR, 2024a.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems. *arXiv preprint arXiv:2402.12875*, 1, 2024b.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Charles London and Varun Kanade. Pause tokens strictly increase the expressivity of constant-depth transformers. *arXiv preprint arXiv:2505.21024*, 2025.

Xutao Ma, Yixiao Huang, Hanlin Zhu, and Somayeh Sojoudi. Breaking the reversal curse in autoregressive language models via identity bridge. *arXiv preprint arXiv:2602.02470*, 2026.

Will Merrill and Ashish Sabharwal. Exact expressive power of transformers with padding. *Advances in Neural Information Processing Systems*, 38:124733–124760, 2025.

William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. *arXiv preprint arXiv:2310.07923*, 2023.

Nikita Nangia and Samuel R. Bowman. ListOps: A diagnostic dataset for latent tree learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 92–99, New Orleans, Louisiana, USA, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-4013.

OpenAI. Learning to reason with LLMs, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International conference on machine learning*, pp. 7487–7498. PMLR, 2020.

Jacob Pfau, William Merrill, and Samuel R Bowman. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Yunwei Ren, Zixuan Wang, and Jason D Lee. Learning and transferring sparse contextual bigrams with linear transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *International Conference on Machine Learning*, pp. 3067–3075. PMLR, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. *arXiv preprint arXiv:2502.03275*, 2025.

Yuandong Tian, Yiping Wang, Beidi Chen, and Simon S Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *Advances in neural information processing systems*, 36:71911–71947, 2023.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023a.

Xinyi Wang, Lucas Caccia, Oleksiy Ostapenko, Xingdi Yuan, William Yang Wang, and Alessandro Sordoni. Guiding language model reasoning with planning tokens. *arXiv preprint arXiv:2310.05707*, 2023b.

Zixuan Wang, Stanley Wei, Daniel Hsu, and Jason D. Lee. Transformers provably learn sparse token selection while fully-connected nets cannot. In *ICML*, 2024.

Zixuan Wang, Eshaan Nichani, Alberto Bietti, Alex Damian, Daniel Hsu, Jason D Lee, and Denny Wu. Learning compositional functions with transformers from easy-to-hard data. *arXiv preprint arXiv:2505.23683*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Kaiyue Wen, Huaqing Zhang, Hongzhou Lin, and Jingzhao Zhang. From sparse dependence to sparse attention: unveiling how chain-of-thought enhances transformer sample efficiency. *arXiv preprint arXiv:2410.05459*, 2024.

Noam Wies, Yoav Levine, and Amnon Shashua. Sub-task decomposition enables learning in sequence to sequence tasks. *arXiv preprint arXiv:2204.02892*, 2022.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. *arXiv preprint arXiv:2409.19606*, 2024a.

Hanlin Zhu, Baihe Huang, Shaolun Zhang, Michael Jordan, Jiantao Jiao, Yuandong Tian, and Stuart J Russell. Towards a theoretical understanding of the'reversal curse'via training dynamics. *Advances in Neural Information Processing Systems*, 37:90473–90513, 2024b.

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Emergence of superposition: Unveiling the training dynamics of chain of continuous thought. *arXiv preprint arXiv:2509.23365*, 2025a.

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. *arXiv preprint arXiv:2505.12514*, 2025b.

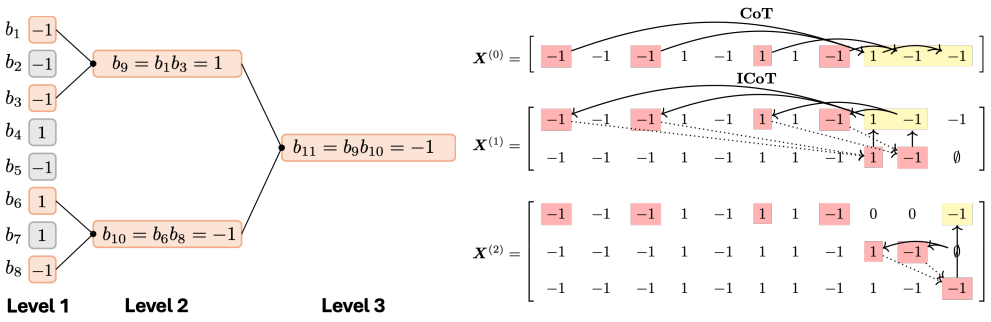

Figure 3: **Illustration of parity learning task with input length $n = 8$ and secret set size $k = 4$.** *Left.* The task can be decomposed into a hierarchical two-parity computation, as specified in Section 3.1. *Right.* Comparison of the training curriculum of Chain-of-Thought (CoT) and Implicit CoT (ICoT). Both methods initially leverage complete thinking traces derived from the hierarchical decomposition. As the ICoT curriculum progresses, these intermediate reasoning steps are replaced by padding tokens (0), forcing the model to internalize the computation within its hidden states.

A Limitations and Future Directions

The analysis concerns parity and a simplified transformer whose depth matches the task hierarchy. It trains only positional attention weights and fixes the link, gates, and value and output maps. The guarantee also uses quantized gradient updates. Our ablations relax several architectural restrictions empirically (Table 2). Extending the theory to learned nonlinearities, content-dependent attention, and trainable gates (Parisotto et al., 2020; Zhu et al., 2024a) remains open.

The curriculum assumes available hierarchical decompositions and intermediate reasoning traces. Our experiments are limited to synthetic tasks and GPT-2 small. Extending Log-ICoT to the practical reasoning tasks studied for ICoT (Deng et al., 2024) requires a way to infer or construct suitable hierarchies.

B Additional Details

B.1 Example of the Parity-Learning Decomposition

To complement the task decomposition in Section 3.1, Figure 3 provides a concrete worked example with $n = 8$ input bits and secret set size $k = 4$. The left panel illustrates the hierarchical decomposition of the k-parity task into a binary tree of two-parity computations. Each internal node b_m $(m > n)$ computes the parity of its two children, with the root b_{11} recovering the target label y. The right panel illustrates the residual-stream evolution $\boldsymbol{X}^{(0)}, \boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}$ across layers, contrasting explicit CoT (top) with the ICoT curriculum (bottom), where intermediate reasoning positions are progressively replaced by padding tokens (0) and the model must internalize the computation in its hidden states.

B.2 Parity Link Functions and Experimental Details

We give the explicit link function from Definition 2 and implementation details for the parity experiments in Section 4.1.

Link functions. The degree-eight link is

$$\phi(x) = -1 + x^2(x^2 - \rho)^2(a + bx^2), \qquad \rho = \tanh^2(1/2), \tag{5}$$

where

$$a = \frac{4(2-\rho)}{(1-\rho)^3}, \qquad b = -\frac{2(3-\rho)}{(1-\rho)^3}.$$

Its quadratic coefficient at zero is $c = a\rho^2 > 0$. The additional stationary decoding points $\pm\tanh(1/2)$ accommodate unequal child logits after quantization in the analysis. For the parity experiments, we use the simpler quartic

$$\phi_{\exp}(x) = -1 + 4x^2 - 2x^4. \tag{6}$$

It maps $[-1, 1]$ into $[-1, 1]$, has $\phi_{\exp}(0) = -1$ and $\phi_{\exp}(\pm 1) = 1$, and its derivative vanishes at 0 and ± 1. It therefore preserves the parity mapping at averages of two signs. The degree-eight link additionally decodes child-logit differences of at most one after integer quantization, as required by Theorem 1.

Model. We use the four-layer model in Definition 3, replacing its link function with the quartic in (6), and retain its causal attention and prescribed gates.

Data generation. Inputs are sampled uniformly from $\{\pm 1\}^n$ with $n = 30$. The secret set $S \subset [n]$ with $|S| = k = 16$ is sampled uniformly from all size-k subsets of $[n]$ at the start of training and held fixed thereafter. The label $y = \prod_{j \in S} b_j$ is computed deterministically from the input. CoT positions $b_{n+1}, \ldots, b_T$ are computed via the recursion $b_m = b_{c_1[m]} \cdot b_{c_2[m]}$ on the parity tree.

Training and evaluation. The quartic-link runs use unquantized parameters and AdamW with attention learning rate 0.1, batch size 500, and 500 updates in each of four stages, totaling one million examples. Current and earlier attention matrices update together. The current and earlier tree levels are padded in the training input, and the loss supervises the current level. The learning rate decays linearly within each stage and resets between stages. Validation pads all CoT positions and averages MSE and accuracy over the current level on $10{,}000$ held-out inputs. All three runs achieve 100% root accuracy on both validation and an independent $10{,}000$-input test set.

B.3 Parity Ablations

The reference is the gated quartic model evaluated in Section 4.1, whose training dynamics and learned attention appear in Figure 2. We vary the prescribed gates, replace the quartic link with a learned MLP, and train either the positional block or the full key-query matrix $\boldsymbol{W}_{\mathsf{KQ}}$. The key map remains the identity, and the value and output maps stay fixed.

All rows use one million examples and the training and evaluation protocol above, with zero-initialized attention. The learned MLP has one hidden ReLU layer of width 512, receives the attention output and query residual, and uses learning rate 0.001. The ungated positional MLP uses attention learning rate 0.03. The ungated full-$\boldsymbol{W}_{\mathsf{KQ}}$ MLP uses $1{,}000$ updates per stage with batch size 250, and attention learning rate 0.1 in the first three stages and 0.03 in the final stage.

Removing the gates while retaining the quartic link (row 2) exposes later layers to near-constant representations. Near-uniform attention averages signs near zero, which the link maps near -1, as described in Section 3.2. Row 3 retains the gates but allows intermediate content to enter the attention scores, coupling attention learning to the evolving representations. This fixed-link model also remains near chance under the tested settings. Replacing the link with a learned MLP resolves the ungated failure empirically with both positional and full key-query training (rows 5 and 6).

Table 2: **Parity ablations.** Final-root validation accuracy (%), mean ± sample standard deviation across three seeds. Each row reports its best measured three-seed setting with current and earlier layers updating.

Link	Gates	W_{KQ} trained	Accuracy
Quartic	yes	positional block	100.000 ± 0.000
Quartic	no	positional block	49.680 ± 0.439
Quartic	yes	full	49.480 ± 1.003
MLP	yes	positional block	99.753 ± 0.215
MLP	no	positional block	100.000 ± 0.000
MLP	no	full	100.000 ± 0.000

B.4 ListOps and Multiplication on GPT-2

ListOps decomposition. Balanced ListOps uses a binary expression tree whose leaves all lie at depth D. Every internal node applies MIN, MAX, MED, or sum modulo 10 to its two children. Level ℓ of the trace therefore holds $2^{D-\ell}$ values, and the root holds the answer. We use $D = 4$ and $D = 5$.

Multiplication reasoning tree. We construct the reasoning tree by recursively splitting both operands into equal-length high and low halves. For a d-digit multiplication, write $a = a_H B + a_L$ and $b = b_H B + b_L$, where $B = 10^{d/2}$. Then

$$ab = B^2(a_H b_H) + B(a_H b_L + a_L b_H) + a_L b_L.$$

The four products define the child subproblems. Repeating this split until each operand has one digit gives a four-ary tree whose leaves are single-digit products. To make the combination at each internal node explicit, let $p_{HH}, p_{HL}, p_{LH}, p_{LL}$ denote its four child products and compute three two-term additions,

$$s = p_{HL} + p_{LH}, \qquad t = p_{LL} + Bs, \qquad p = B^2 p_{HH} + t.$$

For example, 1234×5678 has child subproblems 12×56, 12×78, 34×56, and 34×78, whose products are 672, 936, 1904, and 2652. At the root, $B = 100$, so the two partial sums are $s = 2840$ and $t = 286652$, followed by the answer $p = 7006652$. Each child product is itself computed from four single-digit products.

For 4×4 multiplication, the tree contains 16 single-digit products, four two-digit products, and the root. We serialize the reasoning trace bottom-up in three groups. The first contains the 16 leaf products, the second contains the 12 values (s, t, p) from the four two-digit combinations, and the third contains the root's two partial sums (s, t). The root product is reserved for the final answer. Thus the trace contains 30 intermediate values across three levels, giving four Log-ICoT curriculum stages including initial explicit-CoT training. Values are emitted least-significant digit first, so carries propagate in the generation direction. Intermediate values within each group follow a fixed order, with no explicit level tags.

Model and data. All runs fine-tune the pretrained 12-layer GPT-2 small (Radford et al., 2019) with 124M parameters and its BPE tokenizer, extended with the task's special tokens. Optimization uses AdamW without quantization. Balanced ListOps uses 500,000 training examples per depth, FP32, batch size 32, and a fixed 3,500-update explicit-CoT stage. Multiplication uses the 808,000-example pool of Deng et al. (2024) for all methods, with ICoT runs trained in bf16 at batch size 32 and learning rate 10^{-4}. We use three model seeds per task and method, with the training pool fixed across seeds. We evaluate the final saved checkpoints. On multiplication, each stage's best-validation state is restored before saving.

Curricula. Token-wise ICoT (Deng et al., 2024) removes 4 trace tokens per epoch on depth-4 ListOps and 1 on depth-5 ListOps, with removal smoothing $\lambda = 4$. Smoothing is disabled for multiplication. Log-ICoT removes one tree level per stage, replacing hidden tokens with loss-masked padding and bridging each level transition with 5,000 updates of token-wise removal, without replay. Its ListOps learning rate is 5×10^{-5}. On multiplication, Log-ICoT trains for 5,000 updates after the second tree level is fully hidden before removing the next level.

Stopping. On multiplication, both ICoT methods stop early at intermediate stages once validation accuracy reaches 0.95. The fully implicit stage instead uses plateau stopping. It is evaluated every $5{,}000$ updates and stops after three checks without a 0.005 absolute validation gain once accuracy exceeds 0.90. On ListOps, Log-ICoT stops each level when validation accuracy is within 0.02 of the explicit-CoT baseline, up to a $50{,}000$-update cap. ListOps token-wise ICoT instead uses a fixed removal schedule, with a $25{,}000$-update final recovery at depth 4. Two depth-5 token-wise runs fail the final curriculum gate, which requires fully implicit validation accuracy to be no more than two percentage points below the explicit-CoT baseline. Their fully implicit scores remain in the reported mean. Explicit CoT trains for $10{,}000$ updates on multiplication and generates its own fixed-length trace before the answer marker, without receiving ground-truth intermediate values.

No-padding ablation. Table 3 compares padded Log-ICoT with a variant that deletes hidden trace positions, using the same bridges, learning rate, and no-replay curriculum. The no-padding variant uses a looser stage-transition tolerance and additional final-stage training to accommodate the greater optimization difficulty observed when hidden trace positions are deleted. Its tolerance is 0.1 rather than the padded variant's 0.02, and its final-stage extension is $25{,}000$ updates. It takes more updates at both depths and attains mean validation accuracy 1.6 and 0.7 percentage points higher at depths 4 and 5, respectively. Once the trace is fully internalized, the no-padding variant has the same prompt and completion format as token-wise ICoT, with 63 and 127 prompt tokens at depths 4 and 5, respectively, and 3 completion tokens at both depths. We time the no-padding variant separately. The small latency differences between the two unpadded methods do not indicate a difference in inference workload.

Table 3: **No-padding ablation on ListOps.** Training results are three-seed means $\pm$ sample standard deviations, with updates in thousands, counted up to the final saved checkpoint. Columns match Table 1. No-padding latencies are measured separately using the protocol in Appendix B.5. Speedups are relative to the explicit-CoT rows shown here.

Task	Method	Val. acc. (%)	Updates (10^3)	Latency (ms)	GFLOPs	Speedup
ListOps, $D = 4$	Explicit CoT	99.7 ± 0.1	3.5 ± 0.0	70.4	20.5	$1.0\times$
	Log-ICoT	97.9 ± 0.3	29.8 ± 5.9	10.9	20.5	$6.5\times$
	Log-ICoT, no padding	99.5 ± 0.5	55.6 ± 10.1	10.8	16.2	$6.5\times$
ListOps, $D = 5$	Explicit CoT	100.0 ± 0.0	3.5 ± 0.0	124.4	41.1	$1.0\times$
	Log-ICoT	98.2 ± 0.2	50.9 ± 6.1	10.9	41.2	$11.4\times$
	Log-ICoT, no padding	98.9 ± 0.3	61.1 ± 2.3	10.6	32.5	$11.7\times$

B.5 Inference Cost Details

Protocol. The inference costs in Tables 1 and 3 benchmark prescribed inference formats using pretrained GPT-2 with the same architecture as the fine-tuned models. Each trial times 32 examples after warm-up on one H200, using greedy decoding for exactly the prescribed completion length. The depth-5 ListOps and no-padding measurements average three trials over the same inputs. The other depth-4 ListOps and multiplication rows use one trial per method. The three-trial benchmarks use FP32 GPT-2 with SDPA attention. All speedups use the explicit-CoT baseline reported for the same task.

Dense FLOPs count the attention, projection, MLP and LM-head matrix multiplications of the prefill plus KV-cached decoding. Explicit CoT is priced with its trace on the generated side, since that is what it decodes at inference. The architecture is fixed across methods, so the measurement isolates the format.

C Proof of Section 3

Notations. Recall from Sections 2.1 and 3.1 that $b_1, \ldots, b_n$ are i.i.d. uniform signs and that $b_{n+1}, \ldots, b_T$ are determined by the parity-tree recursion based on the secret set S. For a tuple of indices $(j_1, \ldots, j_r) \in [T]^r$, we say a parity product $b_{j_1} \cdots b_{j_r}$ is *trivial* if it always equals 1. E.g., the parity $b_1 b_3 b_9$ is trivial in Figure 3. We define the set of non-trivial tuples of length r over the first

m indices as:

$$I_{r,m} = \{(j_1, \ldots, j_r) \in [m]^r, b_{j_1} \cdots b_{j_r} \not\equiv 1\}.$$

For $r = 1$, every index is non-trivial so $I_{1,m} = [m]$. As defined in Section 2.2, $\Psi_{t'}^\top \boldsymbol{W}_{\mathsf{O}}^{(t)} = \mathbb{1}(t = t') \left[\mathbf{0}_{B\times T}, \boldsymbol{e}_{L+1,t}^\top \otimes \boldsymbol{I}_B\right] \in \mathbb{R}^{B\times d}$. Let $\boldsymbol{S}_t = \boldsymbol{e}_{L+1,t}^\top \otimes \boldsymbol{I}_B \in \mathbb{R}^{B\times B_L}$, we denote $\boldsymbol{P}_t = \Psi_t^\top \boldsymbol{W}_{\mathsf{O}}^{(t)} = [\mathbf{0}_{B\times T}, \boldsymbol{S}_t]$, which extracts the t-th B-dimensional block. For any $\boldsymbol{x} \in \mathbb{R}^d$, we define the t-th B-block (after the first T coordinates) by

$$\boldsymbol{x}_{[t]} := S_t \boldsymbol{x}_{T:} \in \mathbb{R}^B.$$

Equivalently, we have $(\boldsymbol{x}_{[t]})_i = x_{T+B(t-1)+i}$. Finally, for $x \in \mathbb{R}$ and $S \subseteq \mathbb{R}$, we write $\mathtt{dist}(x, S) := \inf_{s\in S} |x - s|$.

C.1 Representation collapse

Lemma 1 (Representation collapse of input states)**.** *For $B = \mathsf{poly}(n)$, with probability $1 - \exp(-n^{\epsilon/16})$ over random sampling of the input data $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, for all $n^{\epsilon/8} \le m \le n$ we have*

$$\left\| \phi\left(\frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha \right) + \mathbf{1}_B \right\|_\infty \le O(n^{-\epsilon/16}).$$

In particular, when $\boldsymbol{W} = \mathbf{0}_{T\times T}$, attention is uniform and

$$\left\| \phi\left(\sum_{j=1}^{m-1} \sigma_j(\boldsymbol{w}_m) \boldsymbol{b}_j \right) + \mathbf{1}_B \right\|_\infty \le O(n^{-\epsilon/16}).$$

Proof. Set $p_1 = \exp(-n^{\epsilon/16})$ and $\delta_m = p_1/T$. Fix any $n^{\epsilon/8} \le m \le n$. The prefix contains $m-1$ independent input-bit vectors. For any $u > 0$, Hoeffding's inequality and a union bound over the B sample coordinates give

$$\Pr\left(\left\| \frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha \right\|_\infty \ge u \right) \le 2B \exp\left(-\frac{(m-1)u^2}{2} \right).$$

Thus, with probability at least $1 - \delta_m$,

$$\left\| \frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha \right\|_\infty \le \sqrt{\frac{2}{m-1} \log(2TB/p_1)}.$$

Since $\phi(0) = -1$ and $\phi'(0) = 0$, Taylor's theorem gives a constant $C_2 > 0$ depending only on ϕ such that $|\phi(z) + 1| \le C_2 z^2$ for all $z \in [-1, 1]$. Consequently,

$$\begin{aligned} \|\phi(\frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha) + \mathbf{1}_B\|_\infty &= \|\phi(\frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha) - \phi(\mathbf{0}_B)\|_\infty \\ &\le C_2 \frac{2}{m-1} \log(2TB/p_1). \end{aligned}$$

A union bound over at most T prefixes makes this bound simultaneous for all $n^{\epsilon/8} \le m \le n$ with probability at least $1 - p_1$. Recall $B = \mathsf{poly}(n)$ and $T < 2n$. The logarithm then splits as $\log(2TB/p_1) = \log(2TB) + \log(1/p_1) = O(\log n) + n^{\epsilon/16}$, while $m - 1 \ge m/2$ for $m \ge 2$ gives $1/(m-1) \le 2n^{-\epsilon/8}$ for sufficiently large n. Combining the two,

$$C_2 \frac{2}{m-1} \log(2TB/p_1) \;\le\; O\Big(n^{-\epsilon/8} \left(\log n + n^{\epsilon/16} \right) \Big) \;=\; O(n^{-\epsilon/16}),$$

where the last equality holds because $n^{-\epsilon/8} \cdot n^{\epsilon/16} = n^{-\epsilon/16}$ dominates the $n^{-\epsilon/8} \log n$ term. Therefore

$$\|\phi(\frac{1}{m-1} \sum_{\alpha\in[m-1]} \boldsymbol{b}_\alpha) + \mathbf{1}_B\|_\infty \le O(n^{-\epsilon/16}).$$

□

C.2 Residual stream and gradient propagation

For $\ell \in [L]$ and $\beta \in [T]$, denote the causal attention output before the link function by

$$\hat{\boldsymbol{z}}_\beta^{(\ell)} := \sum_{j<\beta} \sigma_j(\boldsymbol{w}_\beta^{(\ell)}) \boldsymbol{x}_j^{(\ell-1)},$$

with the empty-prefix sum equal to $\mathbf{0}_d$. We write $\hat{\boldsymbol{z}}_{\beta,[s]}^{(\ell)}$ for its content block $[s]$. The link ϕ acts coordinatewise.

Lemma 2 (Gated Connection Propagation)**.** *Under the gating weights in Eq. (4), let $\beta \in [T]$ and $\nu := h[\beta]$. The residual stream satisfies*

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \begin{cases} \boldsymbol{x}_{\beta,[1]}^{(0)}, & \nu = 1 \text{ or } 0 \le \ell < \nu - 1, \\ \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)}), & \nu \ge 2 \text{ and } \ell \ge \nu - 1. \end{cases}$$

Proof. Recall the gated connection update at layer ℓ:

$$\boldsymbol{x}_\beta^{(\ell)} = \boldsymbol{x}_\beta^{(\ell-1)} + \boldsymbol{W}_{\mathsf{O}}^{(\ell)} \cdot \mathsf{GC}_\beta^{(\ell)},$$

where $\mathsf{GC}_\beta^{(\ell)} = g_\beta^{(\ell)} \phi(\hat{\boldsymbol{z}}_\beta^{(\ell)}) + (1 - g_\beta^{(\ell)}) \boldsymbol{x}_\beta^{(\ell-1)}$ and $\boldsymbol{W}_{\mathsf{O}}^{(\ell)}$ is the block-shift operator that places content from residual block $[\ell]$ to block $[\ell+1]$.

Componentwise, the block-$[\ell+1]$ update at position β reads

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} + g_\beta^{(\ell)} \phi(\hat{\boldsymbol{z}}_{\beta,[\ell]}^{(\ell)}) + (1 - g_\beta^{(\ell)}) \boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)}, \tag{$\star$}$$

while blocks $[k]$ for $k \neq \ell + 1$ are preserved, i.e., $\boldsymbol{x}_{\beta,[k]}^{(\ell)} = \boldsymbol{x}_{\beta,[k]}^{(\ell-1)}$ for $k \neq \ell + 1$.

Recall the per-block gating from Eq. (4):

$$g_\beta^{(\ell)} = \begin{cases} 1, & n_\ell + 1 \le \beta \le n_{\ell+1}, \\ 0, & \text{otherwise.} \end{cases}$$

Hence $g_\beta^{(\ell)} = 1$ iff $\ell = \nu - 1$ (the writing layer for position β); otherwise $g_\beta^{(\ell)} = 0$.

Case 1: $\nu = 1$ (input position). For $\beta \le n$, $g_\beta^{(\ell)} = 0$ for all ℓ, so Eq. ($\star$) reduces to

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} + \boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)}.$$

We prove by induction on ℓ that $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[1]}^{(0)}$ for all $\ell \ge 0$. The base case $\ell = 0$ holds trivially. For the inductive step, assume $\boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)} = \boldsymbol{x}_{\beta,[1]}^{(0)}$. Block $[\ell+1]$ is unchanged at all earlier layers (since the writing layer for block $[\ell+1]$ is layer ℓ, and earlier layers don't touch block $[\ell+1]$). Hence $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} = \boldsymbol{x}_{\beta,[\ell+1]}^{(0)} = \mathbf{0}_B$ (Section 2.2, the embedding has only block $[1]$ nonzero). Therefore

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \mathbf{0}_B + \boldsymbol{x}_{\beta,[1]}^{(0)} = \boldsymbol{x}_{\beta,[1]}^{(0)}.$$

Case 2: $\nu \ge 2$ and $0 \le \ell < \nu - 1$ (before writing layer). For these layers, $g_\beta^{(\ell)} = 0$, so Eq. ($\star$) again reduces to

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} + \boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)}.$$

By the same induction on ℓ as in Case 1, $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[1]}^{(0)}$ for $0 \le \ell < \nu - 1$. Note that $\boldsymbol{x}_{\beta,[1]}^{(0)}$ here is the embedding content at position β, which can be $\boldsymbol{b}_\beta$ (shown CoT) or $\mathbf{0}_B$ (padded CoT) depending on the curriculum's stage.

Case 3: $\nu \geq 2$ **and** $\ell = \nu - 1$ **(writing layer).** At the writing layer, $g_\beta^{(\nu-1)} = 1$, so Eq. ($\star$) becomes

$$\boldsymbol{x}_{\beta,[\nu]}^{(\nu-1)} = \boldsymbol{x}_{\beta,[\nu]}^{(\nu-2)} + \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)}).$$

Block $[\nu]$ has not been written by any earlier layer (the writing layer for block $[\nu]$ is layer $\nu - 1$, the current layer). Hence $\boldsymbol{x}_{\beta,[\nu]}^{(\nu-2)} = \boldsymbol{x}_{\beta,[\nu]}^{(0)} = \mathbf{0}_B$. Therefore

$$\boldsymbol{x}_{\beta,[\nu]}^{(\nu-1)} = \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)}).$$

Case 4: $\nu \geq 2$ **and** $\ell \geq \nu$ **(after writing layer).** For $\ell \geq \nu$, $g_\beta^{(\ell)} = 0$ (since $\beta \leq n_{\ell-1} < n_\ell + 1$), so Eq. ($\star$) reduces to

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} + \boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)}.$$

We prove by induction on $\ell \geq \nu$ that $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})$. The base case $\ell = \nu - 1$ is Case 3.

For $\ell \geq \nu$: by the inductive hypothesis applied at $\ell - 1 \geq \nu - 1$, $\boldsymbol{x}_{\beta,[\ell]}^{(\ell-1)} = \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})$. Block $[\ell+1]$ has not been written at any earlier layer, so $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell-1)} = \mathbf{0}_B$. Therefore

$$\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \mathbf{0}_B + \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)}) = \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)}).$$

Combining Cases 1, 2 (which give $\boldsymbol{x}_{\beta,[1]}^{(0)}$) and Cases 3, 4 (which give $\phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})$), the lemma follows. These identities also preserve bounded values for every input-bit batch and any initial internal-token contents in $[-1, 1]^B$. At layer zero, the content block is bounded by one. If all block-$[\ell]$ values entering layer ℓ are bounded by one, then their attention average satisfies

$$\|\hat{\boldsymbol{z}}_{\beta,[\ell]}^{(\ell)}\|_\infty \leq \sum_{j<\beta} \sigma_j(\boldsymbol{w}_\beta^{(\ell)}) \|\boldsymbol{x}_{j,[\ell]}^{(\ell-1)}\|_\infty \leq 1;$$

the empty-prefix output is zero. Block $[\ell + 1]$ was zero before this layer writes it. Its new content is either the inherited block-$[\ell]$ value or the link output, both in $[-1, 1]^B$ because $\phi([-1, 1]) \subseteq [-1, 1]$. Induction therefore gives

$$\|\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)}\|_\infty \leq 1, \qquad \|\hat{\boldsymbol{z}}_{\beta,[\ell]}^{(\ell)}\|_\infty \leq 1 \tag{7}$$

at every layer and position, independently of the attention logits. □

Corollary 1 (Frozen Derivative)**.** *Under the setting of Lemma 2, for any $\beta \in [T]$ with $\nu := h[\beta]$, any layer $\ell \in \{0, \ldots, L\}$, and any parameter $w_{j,m}^{(\ell')}$ with $\ell' \in [L]$:*

$$\frac{\partial \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)}}{\partial w_{j,m}^{(\ell')}} = \begin{cases} \dfrac{\partial \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})}{\partial w_{j,m}^{(\ell')}}, & \nu \geq 2,\ \ell \geq \nu - 1,\ \ell' \leq \nu - 1, \\ \mathbf{0}, & \textit{otherwise}. \end{cases}$$

Proof. By Lemma 2, $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)}$ takes two forms:

- If $\nu = 1$, or if $\nu \geq 2$ with $\ell < \nu - 1$: $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \boldsymbol{x}_{\beta,[1]}^{(0)}$. The embedding is data, not a trainable parameter, so the derivative vanishes.
- If $\nu \geq 2$ and $\ell \geq \nu - 1$: $\boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)} = \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})$. By chain rule,

$$\frac{\partial \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)}}{\partial w_{j,m}^{(\ell')}} = \frac{\partial \phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})}{\partial w_{j,m}^{(\ell')}}.$$

  Since $\phi(\hat{\boldsymbol{z}}_{\beta,[\nu-1]}^{(\nu-1)})$ depends only on $\{\boldsymbol{W}^{(\tau)}\}_{\tau \leq \nu-1}$, the derivative vanishes when $\ell' > \nu - 1$, and equals the chain-rule expression when $\ell' \leq \nu - 1$.

In all sub-cases, the derivative matches the corollary's statement. □

Lemma 3 (Per-Node Sensitivity)**.** *Let $L_\phi := \|\phi'\|_\infty$. For any layer indices $\ell' \le \ell \le L$, any query position $m \in [n_{\ell'}+1, n_{\ell'+1}]$, and any $1 \le j \le m-1$, define*

$$\xi^{(\ell)}_{j,m,\ell'} := \max_{\beta: h[\beta] \le \ell+1} \left\| \frac{\partial \boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty .$$

Then

$$\xi^{(\ell)}_{j,m,\ell'} \le 2L_\phi^{\ell-\ell'+1} \sigma_j(\boldsymbol{w}^{(\ell')}_m).$$

Proof. We induct on ℓ.

Base case $\ell = \ell'$. By Corollary 1, the derivative is nonzero only when $\nu \ge 2$, $\ell' \ge \nu - 1$, and $\ell' \le \nu - 1$, i.e., $\nu - 1 = \ell'$, so $\nu = \ell' + 1$. For positions with $\nu > \ell' + 1$, the derivative vanishes since the residual equals the embedding which is not a trainable parameter. For positions with $\nu \le \ell'$, the writing layer is at most $\ell' - 1 < \ell'$, so the derivative also vanishes by Corollary 1.

Hence only β with $\nu = \ell' + 1$ contribute. For such β, Corollary 1 yields

$$\frac{\partial \boldsymbol{x}^{(\ell')}_{\beta,[\ell'+1]}}{\partial w^{(\ell')}_{j,m}} = \frac{\partial \phi(\hat{\boldsymbol{z}}^{(\ell')}_{\beta,[\ell']})}{\partial w^{(\ell')}_{j,m}} = \phi'(\hat{\boldsymbol{z}}^{(\ell')}_{\beta,[\ell']}) \odot \frac{\partial \hat{\boldsymbol{z}}^{(\ell')}_{\beta,[\ell']}}{\partial w^{(\ell')}_{j,m}}.$$

The parameter $w^{(\ell')}_{j,m}$ enters $\hat{\boldsymbol{z}}^{(\ell')}_\beta$ only through the softmax weights at query position $\beta = m$, giving

$$\frac{\partial \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']}}{\partial w^{(\ell')}_{j,m}} = \sigma_j(\boldsymbol{w}^{(\ell')}_m)\big(\boldsymbol{x}^{(\ell'-1)}_{j,[\ell']} - \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']}\big).$$

Each component of $\boldsymbol{x}^{(\ell'-1)}_{j,[\ell']} - \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']}$ lies in $[-2, 2]$, so

$$\xi^{(\ell')}_{j,m,\ell'} \le L_\phi \cdot 2\sigma_j(\boldsymbol{w}^{(\ell')}_m) = 2L_\phi \sigma_j(\boldsymbol{w}^{(\ell')}_m),$$

matching the claim with $\ell - \ell' + 1 = 1$.

Inductive step $\ell \Rightarrow \ell + 1$. Take any $\beta \in [T]$ with $\nu := h[\beta]$. By Corollary 1, the derivative is zero unless $\nu \ge \ell' + 1$ and $\ell + 1 \ge \nu - 1$, i.e., $\nu \le \ell + 2$. So $\nu \in \{\ell' + 1, \dots, \ell + 2\}$.

We split into two sub-cases.

Sub-case A: $\nu \le \ell + 1$. By Lemma 2, both $\boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}$ and $\boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]}$ equal the same value $\phi(\hat{\boldsymbol{z}}^{(\nu-1)}_{\beta,[\nu-1]})$, hence have identical derivatives:

$$\left\| \frac{\partial \boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty = \left\| \frac{\partial \boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty \le \xi^{(\ell)}_{j,m,\ell'}.$$

Sub-case B: $\nu = \ell + 2$. Then β's writing layer is $\ell + 1$, and Corollary 1 gives

$$\frac{\partial \boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}}{\partial w^{(\ell')}_{j,m}} = \frac{\partial \phi(\hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]})}{\partial w^{(\ell')}_{j,m}} = \phi'(\hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}) \odot \frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}}.$$

Since $\ell' \le \ell < \ell + 1$, the parameter $w^{(\ell')}_{j,m}$ does not enter the softmax weights $\sigma_\cdot(\boldsymbol{w}^{(\ell+1)}_\beta)$, so differentiation passes through:

$$\frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} = \sum_{\gamma=1}^{\beta-1} \sigma_\gamma(\boldsymbol{w}^{(\ell+1)}_\beta) \frac{\partial \boldsymbol{x}^{(\ell)}_{\gamma,[\ell+1]}}{\partial w^{(\ell')}_{j,m}}.$$

For an extra causal key with $h[\gamma] > \ell + 1$, Lemma 2 gives $\boldsymbol{x}^{(\ell)}_{\gamma,[\ell+1]} = \boldsymbol{x}^{(0)}_{\gamma,[1]}$. This is input data, so its derivative with respect to $w^{(\ell')}_{j,m}$ is zero. The full-prefix derivative thus reduces to

$$\frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} = \sum_{\substack{\gamma<\beta \\ h[\gamma]\le \ell+1}} \sigma_\gamma(\boldsymbol{w}^{(\ell+1)}_\beta) \frac{\partial \boldsymbol{x}^{(\ell)}_{\gamma,[\ell+1]}}{\partial w^{(\ell')}_{j,m}}.$$

Every derivative in this restricted sum has norm at most $\xi^{(\ell)}_{j,m,\ell'}$ by its definition. The triangle inequality gives

$$\begin{aligned} \left\| \frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty &\le \sum_{\substack{\gamma<\beta \\ h[\gamma]\le \ell+1}} \sigma_\gamma(\boldsymbol{w}^{(\ell+1)}_\beta) \xi^{(\ell)}_{j,m,\ell'} \\ &\le \xi^{(\ell)}_{j,m,\ell'}, \end{aligned} \tag{8}$$

since the restricted attention mass is at most one. Multiplying by $\|\phi'\|_\infty = L_\phi$:

$$\left\| \frac{\partial \boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty \le L_\phi \xi^{(\ell)}_{j,m,\ell'}.$$

Combining sub-cases. Taking the max over β and using $L_\phi \ge 1$,

$$\xi^{(\ell+1)}_{j,m,\ell'} \le L_\phi \xi^{(\ell)}_{j,m,\ell'} \le L_\phi \cdot 2L_\phi^{\ell-\ell'+1} \sigma_j(\boldsymbol{w}^{(\ell')}_m) = 2L_\phi^{(\ell+1)-\ell'+1} \sigma_j(\boldsymbol{w}^{(\ell')}_m).$$

This closes the induction. □

C.3 Bounded Gradient

Fix a sample coordinate $i \in [B]$. Let $\boldsymbol{D}_i(\boldsymbol{b}) \in [-1,1]^{B\times T}$ have i-th row $(\boldsymbol{b}^\top, \boldsymbol{0}^\top_{T-n})$, with arbitrary other rows. Attention depends only on position, the link acts coordinatewise, and the block shifts preserve sample coordinates. Hence the per-example output

$$f^{(t)}_\theta(\boldsymbol{b}) := \boldsymbol{e}^\top_{B,i} \Psi_t^\top \mathcal{T}_\theta(\boldsymbol{D}_i(\boldsymbol{b})) \in \mathbb{R}^T$$

is independent of the other rows and of the choice of i. Let $J^{(t)}_\theta(\boldsymbol{b}) \in \mathbb{R}^{T\times LT^2}$ be its Jacobian with respect to the flattened parameters $\boldsymbol{\theta}$.

Lemma 4 (Bounded Gradient)**.** *For any stage $t \in [L]$, any input $\boldsymbol{b} \in \{\pm 1\}^n$ and any parameters $\boldsymbol{\theta}$,*

$$\|J^{(t)}_{\boldsymbol{\theta}}(\boldsymbol{b})\|_{\mathrm{F}} \le O(n^g),$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, $g = \log_2 L_\phi + 1$ and $L_\phi := \|\phi'\|_\infty$. This bound is used for the final-label comparison in Section 3.3.

Proof. Fix a stage $t \in [L]$. The readout Ψ_t extracts block $[t+1]$, which is unchanged after layer t. By Lemma 2, for any $\alpha \in [T]$ with $h[\alpha] \le t+1$, the readout coordinate is

$$f^{(t)}_{\theta,\alpha}(\boldsymbol{b}) = \left(\boldsymbol{x}^{(t)}_{\alpha,[t+1]}\right)_i,$$

and for α with $h[\alpha] > t+1$, the readout reduces to the embedding due to Lemma 2 (which has zero gradient). It therefore suffices to bound $\|\nabla_\theta f^{(t)}_{\theta,\alpha}(\boldsymbol{b})\|$ at each α with $h[\alpha] \le t+1$, then sum. Fix such an α.

Bounding per-parameter derivatives. For $\ell' > t$, the derivative vanishes by Corollary 1 since the writing layer $h[\alpha] - 1 \le t < \ell'$. At layer $\ell' \in [t]$, only query columns with $n_{\ell'} + 1 \le m \le n_{\ell'+1}$ can have nonzero derivatives, because all other gates are zero. Masked entries with $j \ge m$ also have zero derivative. For $1 \le j < m$ in these columns, Lemma 3 yields

$$\begin{aligned}\left|\frac{\partial f^{(t)}_{\theta,\alpha}(\boldsymbol{b})}{\partial w^{(\ell')}_{j,m}}\right| &= \left|\left(\frac{\partial \boldsymbol{x}^{(t)}_{\alpha,[t+1]}}{\partial w^{(\ell')}_{j,m}}\right)_i\right| \\ &\le \left\|\frac{\partial \boldsymbol{x}^{(t)}_{\alpha,[t+1]}}{\partial w^{(\ell')}_{j,m}}\right\|_\infty \\ &\le \xi^{(t)}_{j,m,\ell'} \le 2L_\phi^{t-\ell'+1}\sigma_j(\boldsymbol{w}^{(\ell')}_m).\end{aligned}$$

Summing over parameters. We compute $\|\nabla_\theta f^{(t)}_{\theta,\alpha}(\boldsymbol{b})\|^2$ by summing the squared derivatives over all trainable parameters. Squaring the per-parameter bound gives

$$\left|\frac{\partial f^{(t)}_{\theta,\alpha}(\boldsymbol{b})}{\partial w^{(\ell')}_{j,m}}\right|^2 \le 4L_\phi^{2(t-\ell'+1)}\sigma_j(\boldsymbol{w}^{(\ell')}_m)^2.$$

Sum over j. Using $\sum_{j=1}^{m-1}\sigma_j(\boldsymbol{w}^{(\ell')}_m)^2 \le \sum_{j=1}^{m-1}\sigma_j(\boldsymbol{w}^{(\ell')}_m) = 1$,

$$\sum_{j=1}^{m-1}\left|\frac{\partial f^{(t)}_{\theta,\alpha}(\boldsymbol{b})}{\partial w^{(\ell')}_{j,m}}\right|^2 \le 4L_\phi^{2(t-\ell'+1)}.$$

Sum over m. There are $n_{\ell'+1} - n_{\ell'} = k \cdot 2^{-\ell'} = O(n)$ query columns with potentially nonzero derivatives at layer ℓ', so

$$\sum_{m,j}\left|\frac{\partial f^{(t)}_{\theta,\alpha}(\boldsymbol{b})}{\partial w^{(\ell')}_{j,m}}\right|^2 \le O(n)\cdot L_\phi^{2(t-\ell'+1)}.$$

Sum over $\ell' \le t$. Since $\phi(1)-\phi(0) = 2$, the mean value theorem gives $L_\phi \ge 2$. Letting $r := t-\ell'+1$ range over $1, 2, \ldots, t$, the geometric sum satisfies

$$\sum_{\ell'=1}^{t}\sum_{m,j}\left|\frac{\partial f^{(t)}_{\theta,\alpha}(\boldsymbol{b})}{\partial w^{(\ell')}_{j,m}}\right|^2 \le \sum_{r=1}^{t} O(n)\cdot L_\phi^{2r} = O(nL_\phi^{2t}).$$

Since $t \le L = \log_2 k$ and $k = O(n)$,

$$O(nL_\phi^{2t}) \le O(nL_\phi^{2L}) = O(n\cdot k^{2\log_2 L_\phi}) = O(n^{2\log_2 L_\phi+1}).$$

Therefore,

$$\|\nabla_\theta f^{(t)}_{\theta,\alpha}(\boldsymbol{b})\|^2 \le O(n^{2\log_2 L_\phi+1}),$$

which gives $\|\nabla_\theta f^{(t)}_{\theta,\alpha}(\boldsymbol{b})\| \le O(n^{\log_2 L_\phi+1/2}) = O(n^{g-1/2})$.

Conclusion. Summing over $\alpha \in [T]$ with $T = O(n)$,

$$\|J^{(t)}_\theta(\boldsymbol{b})\|_{\mathrm{F}}^2 = \sum_\alpha \|\nabla_\theta f^{(t)}_{\theta,\alpha}(\boldsymbol{b})\|_2^2 \le O(n)\cdot O(n^{2g-1}) = O(n^{2g}),$$

giving $\|J^{(t)}_\theta(\boldsymbol{b})\|_{\mathrm{F}} \le O(n^g)$, as claimed. For the scalar root predictor $F_\theta(\boldsymbol{b}) := f^{(L)}_{\theta,T}(\boldsymbol{b})$,

$$\|\nabla_\theta F_\theta(\boldsymbol{b})\|_2 \le \|J^{(L)}_\theta(\boldsymbol{b})\|_{\mathrm{F}} = O(n^g).$$

This is the per-example gradient bound used in the final-label comparison in Appendix D. □

C.4 Preliminary Lemmas

Lemma 5 (Kim & Suzuki (2024), Lemma 9)**.** *Recall that $I_{r,m}$ is defined as the set of non-trivial tuples of length r over the first m indices as:*

$$I_{r,m} = \{(j_1, \ldots, j_r) \in [m]^r, b_{j_1} \cdots b_{j_r} \not\equiv 1\}.$$

Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ be independent input-bit vectors, and define the internal-node vectors by the parity-tree recursion. Samples are independent across the B rows, but the parity columns need not be independent. For any $p > 0$, with probability at least $1 - p$, the following holds for all $r \leq 4$:

$$\max_{(j_1,\ldots,j_r)\in I_{r,m}} \frac{|\langle \boldsymbol{b}_{j_1}, \ldots, \boldsymbol{b}_{j_r}\rangle|}{B} \leq \kappa := \sqrt{\frac{2}{B} \log \frac{64n^4}{p}}.$$

Let $A_j \subseteq [n]$ be the set of input-bit indices whose product defines b_j. For a fixed non-trivial tuple $(j_1, \ldots, j_r)$, the symmetric difference $D := A_{j_1} \triangle \cdots \triangle A_{j_r}$ is nonempty. Its product in sample i is

$$Y_i := \prod_{q=1}^{r} b^i_{j_q} = \prod_{v\in D} b^i_v, \qquad \mathbb{E}[Y_i] = \prod_{v\in D} \mathbb{E}[b^i_v] = 0.$$

The Y_i are independent signs across samples, so Hoeffding's inequality gives

$$\Pr\left(\left|\frac{1}{B}\sum_{i=1}^{B} Y_i\right| > \kappa\right) \leq 2\exp(-B\kappa^2/2).$$

For $T \geq 2$, the number of tuples of lengths one through four is at most

$$\sum_{r=1}^{4} T^r \leq 2T^4 \leq 32n^4.$$

Thus the probability that any non-trivial tuple violates the bound is at most

$$64n^4 \exp(-B\kappa^2/2) = p.$$

This proves the displayed estimate without requiring independence between parity columns. With $B = \Omega(n^{2+\epsilon})$ and $p = \exp(-n^{\epsilon/2})$, it also gives

$$\kappa^2 = \frac{2}{B}\big(\log(64n^4) + n^{\epsilon/2}\big) = O(n^{-2-\epsilon/2}), \qquad \kappa = O(n^{-1-\epsilon/4}).$$

We use this event separately for each fresh stage batch. In particular, we set $B = \Omega(n^{2+\epsilon})$ for some constant $0 < \epsilon \leq 1$ and $p = \exp(-n^{\epsilon/2})$ such that $\kappa = O(n^{-1-\epsilon/4})$. Lemma 5 bounds the contribution of *non-trivial* tuples (those in $I_{r,m}$), where each inner product $|\langle \boldsymbol{b}_{j_1}, \ldots, \boldsymbol{b}_{j_r}\rangle|/B$ is at most κ. The complementary contribution comes from *trivial* tuples (those in $[m]^r \setminus I_{r,m}$), where the inner product equals B deterministically. Bounding such gradient contractions therefore reduces to counting trivial tuples, which we do next for the case $r = 4$.

Lemma 6 (Trivial 4-tuple count)**.** *Let $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \{\pm 1\}^B$ be i.i.d. uniform input bits, and for $\tau \in [n+1, T]$ let $\boldsymbol{b}_\tau := \boldsymbol{b}_{c_1[\tau]} \odot \boldsymbol{b}_{c_2[\tau]}$ denote the parity defined by the tree (Section 3.1). For any $m \in [n+1, T]$, the number of 4-tuples $(\alpha, \beta, \gamma, \delta) \in [m-1]^4$ with $\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta\rangle$ trivial (i.e., $(\alpha, \beta, \gamma, \delta) \notin I_{4,m}$) is at most $O(n^2)$.*

Proof. By the bit-counting argument, $\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta\rangle \equiv B$ happens iff the multiset of input bits in the parity product cancels evenly. Without loss of generality, assume $h[\alpha] \leq h[\beta] \leq h[\gamma] \leq h[\delta]$. Under the causal mask, $\alpha, \beta, \gamma, \delta \in [m-1]$ may take any level, so we enumerate by the level structure.

1. If $h[\beta] < h[\gamma] < h[\delta]$, to cancel out $\boldsymbol{b}_\delta$, it must hold that $\boldsymbol{b}_\gamma$ is a child of $\boldsymbol{b}_\delta$ and $\boldsymbol{b}_\alpha, \boldsymbol{b}_\beta$ are the two children of the other child. This is fully determined by choosing δ and which child is γ, giving $O(n)$ trivial 4-tuples.

2. If $h[\beta] = h[\gamma] < h[\delta]$, then $\boldsymbol{b}_\beta, \boldsymbol{b}_\gamma$ must both be children of $\boldsymbol{b}_\delta$, requiring $h[\gamma] = h[\delta] - 1$. Then $\langle \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta \rangle = B$, and for $\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta \rangle$ to be trivial, we would need $\langle \boldsymbol{b}_\alpha \rangle = B$, which is impossible. No trivial 4-tuples.

3. If $h[\beta] < h[\gamma] = h[\delta]$, then $\gamma = \delta$ (since same-level peers cover disjoint bits, two distinct peers cannot have trivial product). This forces $\alpha = \beta$ similarly. There are $O(n^2)$ such trivial 4-tuples.

4. If $h[\alpha] = h[\beta] = h[\gamma] = h[\delta]$, then the four same-level nodes must pair up to cancel. There are $O(n^2)$ such trivial 4-tuples.

5. If $h[\alpha] < h[\beta] = h[\gamma] = h[\delta]$, then $\langle \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta \rangle$ involves at least $2^{h[\beta]-1}$ distinct input bits, so $\boldsymbol{b}_\alpha$ would need to cover all these bits for the 4-tuple to be trivial. But $\boldsymbol{b}_\alpha$ is a parity over only $2^{h[\alpha]-1} < 2^{h[\beta]-1}$ bits. No trivial 4-tuples.

Combining cases 1, 3, and 4, the total count of trivial 4-tuples is $O(n) + O(n^2) + O(n^2) = O(n^2)$. Cases 2 and 5 contribute none. □

We now proceed to the main proof of Theorem 1.

C.5 Proof of Theorem 1

Definition 4 (Well-trained layer)**.** *For a batch of B inputs, $\boldsymbol{b}_m = (b_m^1, \ldots, b_m^B)^\top \in \{\pm 1\}^B$ denotes the vector of ground-truth parity labels at node m, and $\boldsymbol{x}^{(\ell)}_{m,[\ell+1]} \in \mathbb{R}^B$ denotes the content of residual block $[\ell+1]$ at position m after layer ℓ. Fix a constant $0 < C < 1/5$. Layer $\ell \in [L]$ is* well-trained *if:*

- *(softmax concentration) For all $m \in [n_\ell + 1, n_{\ell+1}]$, every non-child logit $w^{(\ell)}_{j,m}$ with $j < m$ is zero, and*

$$\min_{i \in \{1,2\}} w^{(\ell)}_{c_i[m],m} \geq K_n/5, \qquad \left| w^{(\ell)}_{c_1[m],m} - w^{(\ell)}_{c_2[m],m} \right| \leq 1.$$

  Writing $\mu^{(\ell)}_m := \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}^{(\ell)}_m)$, we also have

$$\mu^{(\ell)}_m \leq \exp(-Cn^{\epsilon/16}), \qquad \frac{1-\mu^{(\ell)}_m}{1+e} \leq \sigma_{c_i[m]}(\boldsymbol{w}^{(\ell)}_m) \leq \frac{e}{1+e}, \quad i \in \{1,2\}.$$

- *(forward error) For every input-bit batch $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \{\pm 1\}^B$ and every choice of initial internal-token contents in $[-1,1]^B$, with all higher residual blocks initialized to zero, for all $m \in [n_\ell + 1, n_{\ell+1}]$, $\|\boldsymbol{x}^{(\ell)}_{m,[\ell+1]} - \boldsymbol{b}_m\|_\infty \leq \exp(-Cn^{\epsilon/16})$.*

A well-trained layer concentrates attention on each node's two correct children and computes their parity with exponentially small error. This is the invariant maintained as the curriculum trains successive layers.

Lemma 7 (Stage 1)**.** *After one approximate-gradient step followed by nearest-integer quantization, starting from zero initialization, on the stage-1 loss $\mathcal{L}^{(1)}$ with B fresh training samples and stage-wise learning rate $\eta_1 = \frac{K_n n^2}{2c}$, where $B = \Omega(n^{2+\epsilon})$, $K_n := \lceil n^{\epsilon/16} \rceil$ and $c > 0$ is the link function constant in Definition 2, with probability $1 - \exp(-n^{\epsilon/2})$ over the sampling of $\boldsymbol{b}^1, \ldots, \boldsymbol{b}^B$, layer 1 is well-trained (Definition 4).*

Proof. At stage 1, the curriculum's input has $\boldsymbol{x}^{(0)}_{\beta,[1]} = \boldsymbol{b}_\beta$ for all $\beta \in [T]$ (full CoT). Recall that

$$\mathcal{L}^{(1)}(\boldsymbol{\theta}) = \frac{1}{2B} \sum_{m=n+1}^{T} \|\Psi_1^\top \boldsymbol{W}_O^{(1)} \left(g_m^{(1)} \phi \left(\sum_{j=1}^{m-1} \sigma_j(\boldsymbol{w}_m^{(1)}) \boldsymbol{x}_j^{(0)} \right) + (1 - g_m^{(1)}) \boldsymbol{x}_m^{(0)} \right) - \boldsymbol{b}_m \|^2.$$

We split into two cases. **Case 1:** $m > n_2$. Layer 1 is inactive ($g_m^{(1)} = 0$); $w_{j,m}^{(1)}$ is structurally decoupled from the loss, giving zero gradient. **Case 2:** $m \in [n+1, n_2]$. Layer 1 is active ($g_m^{(1)} = 1$). Thus, we analyze the gradient signal in detail and treat Case 1 first.

Case 1: $m > n_2$**.** Under per-block gating, $g_m^{(1)} = 0$. The gated connection update at position m becomes

$$\boldsymbol{x}_{m,[2]}^{(1)} = \boldsymbol{x}_{m,[2]}^{(0)} + (1 - g_m^{(1)})\boldsymbol{x}_{m,[1]}^{(0)} + g_m^{(1)}\phi(\hat{\boldsymbol{z}}_m^{(1)}) = \boldsymbol{x}_{m,[2]}^{(0)} + \boldsymbol{x}_{m,[1]}^{(0)},$$

which does not depend on $\hat{\boldsymbol{z}}_m^{(1)}$ and hence not on $\boldsymbol{w}_m^{(1)}$. Therefore $w_{j,m}^{(1)}$ is absent from $\mathcal{L}^{(1)}$, giving $\partial\mathcal{L}^{(1)}/\partial w_{j,m}^{(1)} = 0$ identically. After quantization, $w_{j,m}^{(1)}$ remains zero.

Case 2: $m \in [n+1, n_2]$**.** Put $M := m - 1$, so $n \le M < 2n$. For $m \in [n+1, n_2]$, the loss term reduces to

$$\frac{1}{2B}\|\Psi_1^\top \boldsymbol{W}_{\mathsf{O}}^{(1)}\phi(\hat{\boldsymbol{z}}_m^{(1)}) - \boldsymbol{b}_m\|^2,$$

where $\hat{\boldsymbol{z}}_m^{(1)} = \sum_{j=1}^{M} \sigma_j(\boldsymbol{w}_m^{(1)})\boldsymbol{x}_j^{(0)}$. Standard softmax derivative computations give

$$\frac{\partial \hat{\boldsymbol{z}}_m^{(1)}}{\partial w_{j,m}^{(1)}} = \sigma_j(\boldsymbol{w}_m^{(1)})(\boldsymbol{x}_j^{(0)} - \hat{\boldsymbol{z}}_m^{(1)}).$$

The gradient of $\mathcal{L}^{(1)}$ w.r.t. $w_{j,m}^{(1)}$ is

$$\frac{\partial \mathcal{L}^{(1)}}{\partial w_{j,m}^{(1)}} = \frac{\sigma_j(\boldsymbol{w}_m^{(1)})}{B}\langle \boldsymbol{P}_1^\top(\boldsymbol{P}_1\phi(\hat{\boldsymbol{z}}_m^{(1)}) - \boldsymbol{b}_m), \phi'(\hat{\boldsymbol{z}}_m^{(1)}), \boldsymbol{x}_j^{(0)} - \hat{\boldsymbol{z}}_m^{(1)}\rangle.$$

At initialization $\boldsymbol{W}^{(1)} = 0$, $\sigma_j(\boldsymbol{w}_m^{(1)}) = 1/M$ for $j \in [M]$ and 0 otherwise. Expanding ϕ and ϕ' around 0 via $\phi(t) = -1 + ct^2 + O(t^4)$, $\phi'(t) = 2ct + O(t^3)$,

$$\begin{aligned}
\frac{\partial \mathcal{L}^{(1)}}{\partial w_{j,m}^{(1)}} = \frac{1}{MB}\Bigg(& - \langle \boldsymbol{P}_1^\top \boldsymbol{b}_m, 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle && (9)\\
& + \langle \boldsymbol{P}_1^\top \boldsymbol{P}_1(-\mathbf{1}_d + c\hat{\boldsymbol{z}}_m^2), 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle && (10)\\
& + \langle O(\boldsymbol{P}_1^\top \boldsymbol{P}_1|\hat{\boldsymbol{z}}_m|^4), 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle && (11)\\
& + \langle \boldsymbol{P}_1^\top(\boldsymbol{P}_1\phi(\hat{\boldsymbol{z}}_m) - \boldsymbol{b}_m), O(|\hat{\boldsymbol{z}}_m|^3), \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle \Bigg). && (12)
\end{aligned}$$

Signal term (9)**.** Substituting $\hat{\boldsymbol{z}}_m = \frac{1}{M}\sum_{\alpha\in[M]} \boldsymbol{x}_\alpha$,

$$\begin{aligned}
&\frac{1}{B}\langle \boldsymbol{P}_1^\top \boldsymbol{b}_m, \hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle \\
&= \frac{1}{MB}\left(\sum_{\alpha\in[M]} \langle \boldsymbol{b}_m, \boldsymbol{x}_{\alpha,[1]}, \boldsymbol{x}_{j,[1]}\rangle - \frac{1}{M}\sum_{\alpha,\beta\in[M]} \langle \boldsymbol{b}_m, \boldsymbol{x}_{\alpha,[1]}, \boldsymbol{x}_{\beta,[1]}\rangle \right).
\end{aligned}$$

Since $\boldsymbol{P}_1^\top \boldsymbol{b}_m$ is supported on block 1 only and $\boldsymbol{x}_{\alpha,[1]} = \boldsymbol{b}_\alpha$ at stage 1's input,

$$\langle \boldsymbol{P}_1^\top \boldsymbol{b}_m, \boldsymbol{x}_\alpha, \boldsymbol{x}_\beta\rangle = \langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta\rangle.$$

Using the input-bit supports A_j introduced in the proof of Lemma 5, triviality is equivalent to

$$\boldsymbol{b}_m \odot \boldsymbol{b}_\alpha \odot \boldsymbol{b}_\beta \equiv \mathbf{1}_B \quad \Longleftrightarrow \quad A_\alpha \triangle A_\beta = A_m. \tag{13}$$

If $\alpha < m$ is a same-level peer of m, then $A_\alpha \cap A_m = \varnothing$ and $|A_\alpha| = |A_m|$. The equality would force $A_\beta = A_\alpha \cup A_m$, whose size $2|A_m|$ exceeds that of every node in the prefix. The same argument excludes a same-level β. For the remaining earlier nodes,

$$|A_m| = |A_\alpha \triangle A_\beta| = |A_\alpha| + |A_\beta| - 2|A_\alpha \cap A_\beta| \le |A_m|.$$

Equality forces both supports to have size $|A_m|/2$, to be disjoint, and to have union A_m. In the parity tree these are exactly the supports of $c_1[m]$ and $c_2[m]$. Conversely, the two children have product $\boldsymbol{b}_m$. Hence the only trivial ordered pairs are $(c_1[m], c_2[m])$ and $(c_2[m], c_1[m])$. For non-trivial tuples, Lemma 5 gives $|\langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta \rangle| \le B\kappa$ where $\kappa = O(n^{-1-\epsilon/4})$. Combining,

$$\frac{1}{B} \sum_{\alpha,\beta \in [M]} \langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta \rangle = 2 + O(n^2\kappa).$$

Similarly, $\langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_j \rangle$ is trivial only when $p[j] = m$ and α is the other child of m. As a result,

$$\frac{1}{B} \sum_{\alpha \in [M]} \langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_j \rangle = \begin{cases} 1 + O(n\kappa) & p[j] = m, \\ O(n\kappa) & \text{otherwise.} \end{cases}$$

Combining the two sums, including their two ordered-child contributions,

$$\begin{aligned} -\frac{1}{MB} \langle \boldsymbol{P}_1^\top \boldsymbol{b}_m, 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m \rangle &= -\frac{2c}{M^2} \left(\mathbb{1}(p[j] = m) + O(M\kappa) - \frac{2 + O(M^2\kappa)}{M} \right) \\ &= -\frac{2c}{M^2} \mathbb{1}(p[j] = m) + \frac{4c}{M^3} + O(\kappa/M) \\ &= -\frac{2c}{M^2} \mathbb{1}(p[j] = m) + O(n^{-2-\epsilon/4}). \end{aligned}$$

Here $M \ge n$ gives $M^{-3} = O(n^{-3})$ and $\kappa/M = O(n^{-2-\epsilon/4})$, and $0 < \epsilon \le 1$ makes the former no larger than the latter order. Next for Eq. (10), we write

$$\begin{aligned} &\frac{1}{B} \langle \boldsymbol{P}_1^\top \boldsymbol{P}_1(-\mathbf{1}_d + c\hat{\boldsymbol{z}}_m^2), 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m \rangle \\ &= -\frac{2c}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}, \boldsymbol{x}_{j,[1]} \rangle + \frac{2c}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^2 \rangle + \frac{2c^2}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^3, \boldsymbol{x}_{j,[1]} \rangle - \frac{2c^2}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^4 \rangle. \end{aligned}$$

For the second-order terms, we have:

$$\begin{aligned} \frac{1}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}, \boldsymbol{x}_{j,[1]} \rangle &= \frac{1}{MB} \left(\langle \boldsymbol{b}_j, \boldsymbol{b}_j \rangle + \sum_{\alpha \ne j,\, \alpha \in [M]} \langle \boldsymbol{b}_\alpha, \boldsymbol{b}_j \rangle \right) = \frac{1}{M} + O(\kappa), \\ \frac{1}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^2 \rangle &= \frac{1}{M^2 B} \left(\sum_{\alpha \in [M]} \langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\alpha \rangle + \sum_{\alpha \ne \beta,\, \alpha,\beta \in [M]} \langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta \rangle \right) = \frac{1}{M} + O(\kappa). \end{aligned}$$

For the fourth-order interaction terms, we discuss when $(\alpha, \beta, \gamma, \delta) \notin I_{4,m}$. By Lemma 6, the total count of trivial 4-tuples is $O(n^2)$. Therefore,

$$\begin{aligned} \frac{1}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^4 \rangle &= \frac{1}{M^4 B} \sum_{\alpha,\beta,\gamma,\delta \in [M]} \langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta \rangle \\ &= \frac{1}{M^4 B} \left(\sum_{\alpha,\beta,\gamma,\delta \in I_{4,m}} O(B\kappa) + \sum_{\alpha,\beta,\gamma,\delta \notin I_{4,m}} B \right) \\ &\le \frac{1}{M^4} \left(O(n^2) + M^4\kappa \right) \\ &= O(n^{-2} + \kappa). \end{aligned}$$

Now for $\langle \hat{\boldsymbol{z}}_{m,[1]}^3, \boldsymbol{x}_{j,[1]} \rangle$, assuming index j is contained in $(\alpha, \beta, \gamma, \delta)$. Then case 1 in Lemma 6 has $O(1)$ trivial tuples, cases 3 and 4 are reduced to $O(n)$ (one free index). Taking absolute values before applying the tuple bounds gives

$$\begin{aligned} \left| \frac{1}{B} \langle \hat{\boldsymbol{z}}_{m,[1]}^3, \boldsymbol{x}_{j,[1]} \rangle \right| &\le \frac{1}{M^3 B} \sum_{\alpha,\beta,\gamma \in [M]} |\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_j \rangle| \\ &\le \frac{O(n)B + M^3 B\kappa}{M^3 B} = O(n^{-2} + \kappa). \end{aligned} \tag{14}$$

Here trivial terms have magnitude B, and each remaining term has magnitude at most $B\kappa$ by Lemma 5. Combining,

$$\begin{aligned}
&\frac{1}{MB}\langle \boldsymbol{P}_1^\top \boldsymbol{P}_1(-\mathbf{1}_d + c\hat{\boldsymbol{z}}_m^2), 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle \\
&= \frac{1}{M}\left(-2c\left(\frac{1}{M} + O(\kappa)\right) + 2c\left(\frac{1}{M} + O(\kappa)\right) + O(n^{-2} + \kappa)\right) \\
&= O(M^{-3} + \kappa/M) = O(n^{-2-\epsilon/4}).
\end{aligned}$$

For Eq. (11), note that

$$\frac{1}{B}\langle \boldsymbol{P}_1^\top \boldsymbol{P}_1 |\hat{\boldsymbol{z}}_m|^4\rangle = \frac{1}{B}\langle \hat{\boldsymbol{z}}_{m,[1]}^4\rangle = O(n^{-2} + \kappa),$$

and since $2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m$ are contained in $[-2c, 2c]$ and $[-2, 2]$ respectively,

$$\frac{1}{MB}\langle O(\boldsymbol{P}_1^\top \boldsymbol{P}_1 |\hat{\boldsymbol{z}}_m|^4), 2c\hat{\boldsymbol{z}}_m, \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle = \frac{4c}{M}O(n^{-2} + \kappa) = O(n^{-2-\epsilon/4}).$$

Finally for Eq. (12), by Cauchy-Schwarz,

$$\begin{aligned}
\frac{1}{B}\langle \boldsymbol{P}_1^\top \boldsymbol{P}_1 |\hat{\boldsymbol{z}}_m|^3\rangle = \frac{1}{B}\sum_{i=1}^{B} |\hat{\boldsymbol{z}}_{m,[1],i}^3| &\le \frac{1}{B}\left(\sum_{i=1}^{B} \hat{\boldsymbol{z}}_{m,[1],i}^2\right)^{1/2}\left(\sum_{i=1}^{B} \hat{\boldsymbol{z}}_{m,[1],i}^4\right)^{1/2} \\
&= O\Big((M^{-1} + \kappa)^{1/2}(n^{-2} + \kappa)^{1/2}\Big) = O(n^{-1-\epsilon/8}).
\end{aligned}$$

By the definition of $\boldsymbol{P}_1$,

$$\begin{aligned}
&\frac{1}{MB}\langle \boldsymbol{P}_1^\top(\boldsymbol{P}_1\phi(\hat{\boldsymbol{z}}_m) - \boldsymbol{b}_m), O(|\hat{\boldsymbol{z}}_m|^3), \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle \\
&= \frac{1}{MB}\langle \phi(\hat{\boldsymbol{z}}_m) - \boldsymbol{P}_1^\top \boldsymbol{b}_m, O(\boldsymbol{P}_1^\top \boldsymbol{P}_1 |\hat{\boldsymbol{z}}_m|^3), \boldsymbol{x}_j - \hat{\boldsymbol{z}}_m\rangle \\
&= \frac{4}{MB}\langle O(\boldsymbol{P}_1^\top \boldsymbol{P}_1 |\hat{\boldsymbol{z}}_m|^3)\rangle \\
&= O(n^{-2-\epsilon/8}).
\end{aligned}$$

Combining all terms from (9) to (12), for any $j \in [M]$, we get

$$\frac{\partial \mathcal{L}^{(1)}}{\partial w_{j,m}^{(1)}} = -\frac{2c}{M^2}\mathbb{1}(p[j] = m) + O(n^{-2-\epsilon/8}).$$

Note that this applies to the approximate gradient $\tilde{\nabla}_{w_{j,m}^{(1)}}\mathcal{L}^{(1)}$ too since each component of the noise is bounded by $O(n^{-2-\epsilon/8})$.

Concentration of attention scores at $m \in [n+1, n_2]$. The gradient has child signal $2c/M^2$ and a remainder of order $n^{-2-\epsilon/8}$. We retain the original stage-wise learning rate

$$\eta_1 := \frac{K_n n^2}{2c}, \qquad K_n := \lceil n^{\epsilon/16}\rceil,$$

but the unrounded child update now depends on the query:

$$\lambda_m := K_n\frac{n^2}{M^2}, \qquad K_n/4 \le \lambda_m \le K_n.$$

The update becomes

$$w_{j,m}^{(1)}(1) = q(\lambda_m \mathbb{1}(p[j] = m) + \Delta_{j,m}), \tag{15}$$

where $\Delta_{j,m} := \eta_1 R_{j,m}$ and $R_{j,m}$ is the remainder in the negative approximate gradient, including the coordinatewise oracle error. Uniformly over $j < m$,

$$|R_{j,m}| \le C_\nabla n^{-2-\epsilon/8},$$

and, since $K_n \le 2n^{\epsilon/16}$,

$$|\Delta_{j,m}| \le \frac{C_\nabla K_n n^2}{2c}\, n^{-2-\epsilon/8} \le \frac{C_\nabla}{c} n^{-\epsilon/16} < 1/4 \tag{16}$$

for sufficiently large n.

Non-child indices. For $p[j] \neq m$, the update is $q(\Delta_{j,m}) = 0$ because $|\Delta_{j,m}| < 1/4 < 1/2$.

Children indices. For $i \in \{1, 2\}$, write

$$u_i := \lambda_m + \Delta_{c_i[m],m}, \qquad a_i := w^{(1)}_{c_i[m],m}(1) = q(u_i).$$

Nearest-integer rounding satisfies $|q(u) - u| \le 1/2$ for every $u \in \mathbb{R}$. Therefore,

$$|a_i - \lambda_m| \le |q(u_i) - u_i| + |\Delta_{c_i[m],m}| < \frac{1}{2} + \frac{1}{4} = \frac{3}{4}.$$

Using $\lambda_m \ge K_n/4$, we obtain

$$\min_{i\in\{1,2\}} a_i \ge \lambda_m - \frac{3}{4} \ge \frac{K_n}{4} - \frac{3}{4} \ge \frac{K_n}{5},$$

where the last inequality holds once $K_n \ge 15$, as is the case for sufficiently large n. Moreover, the two unrounded updates satisfy

$$|u_1 - u_2| = |\Delta_{c_1[m],m} - \Delta_{c_2[m],m}| \le |\Delta_{c_1[m],m}| + |\Delta_{c_2[m],m}| < \frac{1}{2}.$$

Hence

$$|a_1 - a_2| \le |a_1 - u_1| + |u_1 - u_2| + |u_2 - a_2| < \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{3}{2}.$$

Since $a_1 - a_2 \in \mathbb{Z}$, this implies $|a_1 - a_2| \le 1$. In the original notation,

$$\min_{i\in\{1,2\}} w^{(1)}_{c_i[m],m}(1) \ge K_n/5, \qquad \left|w^{(1)}_{c_1[m],m}(1) - w^{(1)}_{c_2[m],m}(1)\right| \le 1. \tag{17}$$

Softmax concentration. There are $M - 2$ non-child keys, all with zero logits. With the child logits a_1, a_2 defined above,

$$\begin{aligned}\mu^{(1)}_m := \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}^{(1)}_m) &= \frac{M-2}{e^{a_1} + e^{a_2} + M - 2} \\ &\le \frac{M-2}{2e^{K_n/5} + M - 2} \le 2ne^{-K_n/5}.\end{aligned} \tag{18}$$

Since $K_n \ge n^{\epsilon/16}$ and $\log(2n) \le (1/5 - C)n^{\epsilon/16}$ for sufficiently large n,

$$2ne^{-K_n/5} \le \exp\left(\log(2n) - \frac{1}{5}n^{\epsilon/16}\right) \le \exp(-Cn^{\epsilon/16}).$$

For $i \in \{1, 2\}$, the exact child probability is

$$\sigma_{c_i[m]}(\boldsymbol{w}^{(1)}_m) = (1 - \mu^{(1)}_m)\frac{e^{a_i}}{e^{a_1} + e^{a_2}} = \frac{1 - \mu^{(1)}_m}{1 + e^{a_{3-i} - a_i}}.$$

The bound $|a_1 - a_2| \le 1$ gives $e^{-1} \le e^{a_{3-i}-a_i} \le e$, and therefore

$$\frac{1 - \mu^{(1)}_m}{1 + e} \le \sigma_{c_i[m]}(\boldsymbol{w}^{(1)}_m) \le \frac{e(1 - \mu^{(1)}_m)}{1 + e} \le \frac{e}{1 + e}. \tag{19}$$

Forward pass evaluation at $m \in [n+1, n_2]$. For $m \in [n+1, n_2]$, the children $c_1[m], c_2[m]$ are at level 1 (input bits), so $\boldsymbol{x}^{(0)}_{c_i[m],[1]} = \boldsymbol{b}_{c_i[m]}$ exactly. Normalize only over the two children:

$$\theta_m := \frac{e^{w^{(1)}_{c_1[m],m}}}{e^{w^{(1)}_{c_1[m],m}} + e^{w^{(1)}_{c_2[m],m}}}, \qquad \boldsymbol{s}_m := \theta_m \boldsymbol{b}_{c_1[m]} + (1-\theta_m)\boldsymbol{b}_{c_2[m]}.$$

Put $d_m := a_1 - a_2$. Quantization and (17) give $d_m \in \{-1, 0, 1\}$. Consequently,

$$2\theta_m - 1 = \frac{e^{d_m} - 1}{e^{d_m} + 1} = \tanh(d_m/2) \in \{-r, 0, r\}. \tag{20}$$

In sample i, if $b^i_{c_1[m]} = b^i_{c_2[m]}$, then $(\boldsymbol{s}_m)_i = b^i_{c_1[m]} \in \{\pm 1\}$ and $b^i_m = 1$. If the child signs differ, then $(\boldsymbol{s}_m)_i = (2\theta_m - 1)b^i_{c_1[m]} \in \{-r, 0, r\}$ and $b^i_m = -1$. The values and stationary points in Definition 2 thus give $\phi(\boldsymbol{s}_m) = \boldsymbol{b}_m$ and $\phi'(\boldsymbol{s}_m) = \boldsymbol{0}_B$.

All attention values lie in $[-1, 1]^B$, including arbitrary bounded initial internal-token contents. Since $\sigma_{c_1[m]} = (1 - \mu^{(1)}_m)\theta_m$ and $\sigma_{c_2[m]} = (1 - \mu^{(1)}_m)(1 - \theta_m)$, their contribution is $(1 - \mu^{(1)}_m)\boldsymbol{s}_m$. Thus

$$\hat{\boldsymbol{z}}^{(1)}_{m,[1]} - \boldsymbol{s}_m = \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}^{(1)}_m)\boldsymbol{x}^{(0)}_{j,[1]} - \mu^{(1)}_m \boldsymbol{s}_m. \tag{21}$$

Taking norms gives

$$\begin{aligned}\|\hat{\boldsymbol{z}}^{(1)}_{m,[1]} - \boldsymbol{s}_m\|_\infty &\le \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}^{(1)}_m)\|\boldsymbol{x}^{(0)}_{j,[1]}\|_\infty + \mu^{(1)}_m \|\boldsymbol{s}_m\|_\infty \\ &\le 2\mu^{(1)}_m \le 2\exp(-Cn^{\epsilon/16}).\end{aligned}$$

Let $\mathcal{Z} := \{-1, -r, 0, r, 1\}$ be the stationary decoding set from Definition 2. For scalars $v, s \in [-1, 1]$ with $s \in \mathcal{Z}$, Taylor's theorem with integral remainder gives

$$\phi(v) - \phi(s) = (v - s)\phi'(s) + (v - s)^2 \int_0^1 (1 - u)\phi''(s + u(v - s))\, du.$$

Since $\phi'(s) = 0$ and $\int_0^1 (1 - u)\, du = 1/2$, setting $C_2 := \frac{1}{2}\sup_{|u|\le 1} |\phi''(u)|$ yields

$$|\phi(v) - \phi(s)| \le C_2 |v - s|^2. \tag{22}$$

Applying this coordinatewise at $\boldsymbol{s}_m$ gives

$$\begin{aligned}\|\boldsymbol{x}^{(1)}_{m,[2]} - \boldsymbol{b}_m\|_\infty &= \|\phi(\hat{\boldsymbol{z}}^{(1)}_{m,[1]}) - \phi(\boldsymbol{s}_m)\|_\infty \\ &\le C_2\big(2\exp(-Cn^{\epsilon/16})\big)^2 \le \exp(-Cn^{\epsilon/16}),\end{aligned} \tag{23}$$

The last inequality holds once $4C_2 \exp(-Cn^{\epsilon/16}) \le 1$. Once the logit bounds hold, these forward estimates use no sampled correlations, so this proves the uniform forward bound in Definition 4.

□

At each stage $t \ge 1$, we replace the first $k(1 - 2^{-(t-1)})$ thinking tokens with padding tokens. This ensures that at $L := \log_2 k$ stages, we are relying only on the input bits to predict. The input sequence looks like

$$\boldsymbol{b}^{(t)} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n, \underbrace{\boldsymbol{0}_B, \ldots, \boldsymbol{0}_B}_{n_t - n}, \boldsymbol{b}_{n_t+1}, \boldsymbol{b}_{n_t+2}, \ldots, \boldsymbol{b}_T)$$

The loss function is only computed on position $n_t + 1 \le m \le T$, i.e.,

$$\mathcal{L}^{(t)}(\boldsymbol{\theta}) = \frac{1}{2B} \sum_{m=n_t+1}^{T} \|\hat{f}(\boldsymbol{D})_m - \boldsymbol{b}_m\|^2$$

At training time, since the first $k(1 - 2^{-(t-1)})$ thinking tokens are masked, the model needs to rely on the hidden thinking traces in t-th layer to do the prediction for $n_t + 1 \le m \le n_{t+1}$. Similar to the first stage, we can learn the t-th layer by one step of gradient descent as shown in Lemma 9. Before proceeding, we first state a lemma that characterizes how the hidden states progress over layers.

Lemma 8 (Characterization of Hidden States at Stage s)**.** *Fix a curriculum stage $s \in [L]$. Suppose that for all $\tau \in \{1, \dots, s-1\}$, the per-layer recursion satisfies*

$$\boldsymbol{x}^{(\tau)}_{m,[\tau+1]} = \begin{cases} \boldsymbol{b}_m + \boldsymbol{\xi}, & n_\tau + 1 \le m \le n_{\tau+1}, \\ \boldsymbol{x}^{(\tau-1)}_{m,[\tau]}, & \textit{otherwise}, \end{cases} \tag{24}$$

where $\|\boldsymbol{\xi}\|_\infty \le \delta := \exp(-Cn^{\epsilon/16})$. Then for any layer $\ell \in \{0, 1, \dots, s-1\}$ and any $\beta \in [T]$,

$$\boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]} = \begin{cases} \boldsymbol{b}_\beta + \boldsymbol{\xi}, & n+1 \le \beta \le n_{\ell+1}, \\ \boldsymbol{b}_\beta, & \beta \le n \textit{ or } \beta > n_s, \\ \boldsymbol{0}_B, & n_{\ell+1} < \beta \le n_s, \end{cases} \tag{25}$$

Proof. We prove Eq. (25) by induction on ℓ.

Base case ($\ell = 0$). At $\ell = 0$, the residual stream is the curriculum's stage-s input:

$$\boldsymbol{x}^{(0)}_{\beta,[1]} = \begin{cases} \boldsymbol{b}_\beta, & \beta \le n \text{ or } \beta > n_s, \\ \boldsymbol{0}_B, & n+1 \le \beta \le n_s. \end{cases}$$

Since $n_{\ell+1} = n_1 = n$, the "computed" range $n+1 \le \beta \le n_{\ell+1}$ is empty, and the three cases above coincide with the three cases of Eq. (25) at $\ell = 0$.

Inductive step ($\ell \Rightarrow \ell+1$, for $\ell + 1 \le s-1$). Assume Eq. (25) holds at layer ℓ. We verify it at layer $\ell+1$ by partitioning $[T]$ into two regions and applying Eq. (24) at $\tau = \ell + 1$:

Region 1: Newly computed positions ($n_{\ell+1} + 1 \le \beta \le n_{\ell+2}$). By Eq. (24),

$$\boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]} = \boldsymbol{b}_\beta + \boldsymbol{\xi}, \quad \|\boldsymbol{\xi}\|_\infty \le \delta.$$

Since $n+1 \le \beta \le n_{\ell+2}$, this matches the computed-range case of Eq. (25) at $\ell+1$.

Region 2: Inherited positions ($\beta \le n_{\ell+1}$ or $\beta > n_{\ell+2}$). For these positions, Eq. (24) gives $\boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]} = \boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]}$, so the value is inherited from layer ℓ and we apply the inductive hypothesis:

- If $n+1 \le \beta \le n_{\ell+1}$ (previously computed), then $\boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]} = \boldsymbol{b}_\beta + \boldsymbol{\xi}$ with $\|\boldsymbol{\xi}\|_\infty \le \delta$. Since $\beta \in [n+1, n_{\ell+2}]$, this matches the computed-range case at $\ell+1$.
- If $\beta \le n$ or $\beta > n_s$ (shown), then $\boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]} = \boldsymbol{b}_\beta$, matching the shown-range case at $\ell+1$.
- If $n_{\ell+2} < \beta \le n_s$ (padded), then β also satisfies $n_{\ell+1} < \beta \le n_s$, so by the inductive hypothesis $\boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]} = \boldsymbol{0}_B$, matching the padded-range case at $\ell+1$.

In both regions, Eq. (25) holds at layer $\ell+1$. This completes the induction. □

Lemma 9 (Inductive Step)**.** *Fix any $t \in \{1, \dots, L-1\}$. Work at the stage-t iterate of Algorithm 1. Suppose layers $1, \dots, t$ are well-trained (Definition 4) and $\boldsymbol{W}^{(\ell)}(t) = 0$ for all $\ell \ge t+1$. After one approximate-gradient step followed by nearest-integer quantization on the stage-$(t+1)$ loss $\mathcal{L}^{(t+1)}$ with B training samples and stage-wise learning rate $\eta_{t+1} = K_n n_{t+1}^2/(2c)$, where $B = \Omega(n^{2+\epsilon})$, $K_n := \lceil n^{\epsilon/16} \rceil$ and $c > 0$ is the link function constant in Definition 2, with probability $1 - \exp(-n^{\epsilon/2})$ over the fresh stage-$(t+1)$ batch, layer $t+1$ is also well-trained (Definition 4).*

Proof. By the assumption that layers $1, \dots, t$ are well-trained and Lemma 8 (applied with $s = t+1$, $\ell = t$), the hidden states satisfy $\boldsymbol{x}^{(t)}_{\tau,[t+1]} = \boldsymbol{b}_\tau + \boldsymbol{\xi}$ with $\|\boldsymbol{\xi}\|_\infty \le \delta = \exp(-Cn^{\epsilon/16})$ for all $\tau \in [T]$. Condition on the batches through stage t, so the current weights are fixed. The stage-$(t+1)$ batch is independent of this history. Applying Lemma 5 to that fresh batch gives the same correlation event

with conditional failure probability at most $\exp(-n^{\epsilon/2})$. For scalars $y_1, \dots, y_r \in [-1, 1]$ and exact signs $b_1, \dots, b_r$ satisfying $|y_q - b_q| \le \delta$, the telescoping identity is

$$\prod_{q=1}^{r} y_q - \prod_{q=1}^{r} b_q = \sum_{q=1}^{r} (y_q - b_q) \left(\prod_{v<q} y_v \right) \left(\prod_{v>q} b_v \right).$$

Each product multiplying $y_q - b_q$ has absolute value at most one. Hence

$$\left| \prod_{q=1}^{r} y_q - \prod_{q=1}^{r} b_q \right| \le \sum_{q=1}^{r} |y_q - b_q| \le r\delta. \tag{26}$$

The hidden values are bounded by (7), so this estimate applies samplewise to every product below. In particular, replacing up to four hidden values by their exact parity signs changes the corresponding empirical inner product by at most $4B\delta$. Recall that

$$\mathcal{L}^{(t+1)}(\boldsymbol{\theta})$$
$$= \frac{1}{2B} \sum_{m=n_{t+1}+1}^{T} \|\Psi_{t+1}^{\top} \boldsymbol{W}_{\mathsf{O}}^{(t+1)} \left(g_m^{(t+1)} \phi \left(\sum_{j=1}^{m-1} \sigma_j(\boldsymbol{w}_m^{(t+1)}) \boldsymbol{x}_j^{(t)} \right) + (1 - g_m^{(t+1)}) \boldsymbol{x}_m^{(t)} \right) - \boldsymbol{b}_m \|^2.$$

We split into two cases.

Case 1: $m \in [n_{t+1} + 1, n_{t+2}]$**.** Here $g_m^{(t+1)} = 1$ (active) and the gradient encodes a non-trivial signal from the contraction.

Case 2: $m > n_{t+2}$**.** Here $g_m^{(t+1)} = 0$ by per-block gating, layer $t + 1$ does not write at position m, and the gradient is identically zero. We treat case 1 in detail and then briefly address case 2 at the end of the proof.

Case 1: $m \in [n_{t+1}+1, n_{t+2}]$**.** Put $M := m-1$. At zero initialization of layer $t+1$, $\sigma_j(\boldsymbol{w}_m) = 1/M$ for all $j < m$ and is zero otherwise. In particular, the prefix includes earlier peers at level $t + 2$. Those peers are still displayed in the stage-$(t + 1)$ input, so their block-$[t + 1]$ values equal their parity vectors exactly; the lower-level values differ from their parity vectors by at most δ.

$$\begin{aligned}
\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(t+1)}} =& \frac{\sigma_j(\boldsymbol{w}_m^{(t+1)})}{B} \langle \boldsymbol{P}_{t+1}^{\top} (\boldsymbol{P}_{t+1} \phi(\hat{\boldsymbol{z}}_m^{(t+1)}) - \boldsymbol{b}_m), \phi'(\hat{\boldsymbol{z}}_m^{(t+1)}), \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle \\
=& -\frac{1}{MB} \langle \boldsymbol{P}_{t+1}^{\top} \boldsymbol{b}_m, 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle && (27) \\
&+ \frac{1}{MB} \langle \boldsymbol{P}_{t+1}^{\top} \boldsymbol{P}_{t+1} (-\mathbf{1}_d + c(\hat{\boldsymbol{z}}_m^{(t+1)})^2), 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle && (28) \\
&+ \frac{1}{MB} \langle O(\boldsymbol{P}_{t+1}^{\top} \boldsymbol{P}_{t+1} |\hat{\boldsymbol{z}}_m^{(t+1)}|^4), 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle && (29) \\
&+ \frac{1}{MB} \langle \boldsymbol{P}_{t+1}^{\top} (\boldsymbol{P}_{t+1} \phi(\hat{\boldsymbol{z}}_m^{(t+1)}) - \boldsymbol{b}_m), O(|\hat{\boldsymbol{z}}_m^{(t+1)}|^3), \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle. && (30)
\end{aligned}$$

For Equation (27), we substitute $\hat{\boldsymbol{z}}_m^{(t+1)} = \frac{1}{M} \sum_{\alpha \in [M]} \boldsymbol{x}_\alpha^{(t)}$ to expand

$$\begin{aligned}
&\frac{1}{B} \langle \boldsymbol{P}_{t+1}^{\top} \boldsymbol{b}_m, \hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)} \rangle \\
&= \frac{1}{MB} \left(\sum_{\alpha \in [M]} \langle \boldsymbol{b}_m, \boldsymbol{x}_{\alpha,[t+1]}^{(t)}, \boldsymbol{x}_{j,[t+1]}^{(t)} \rangle - \frac{1}{M} \sum_{\alpha,\beta \in [M]} \langle \boldsymbol{b}_m, \boldsymbol{x}_{\alpha,[t+1]}^{(t)}, \boldsymbol{x}_{\beta,[t+1]}^{(t)} \rangle \right).
\end{aligned}$$

The support argument in (13) applies at every tree level: earlier same-level peers are disjoint from m, and all other prefix nodes have support size at most half that of m. Therefore the only trivial ordered pairs remain $(c_1[m], c_2[m])$ and $(c_2[m], c_1[m])$. For the two ordered child pairs,

$$\langle \boldsymbol{b}_m, \boldsymbol{x}_{\alpha,[t+1]}^{(t)}, \boldsymbol{x}_{\beta,[t+1]}^{(t)} \rangle = \langle \boldsymbol{b}_m, \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta \rangle + O(\delta B) = B + O(\delta B),$$

and for non-trivial (α, β),

$$\left|\langle \boldsymbol{b}_m, \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{\beta,[t+1]}\rangle\right| \le B\kappa + 2\delta B = O(B(\kappa+\delta)).$$

Using $M = \Theta(n)$ (specifically $n \le M < 2n$), putting these together,

$$\frac{1}{B}\sum_{\alpha,\beta\in[M]} \langle \boldsymbol{b}_m, \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{\beta,[t+1]}\rangle = 2 + O(n^2\delta) + O(n^2\kappa) = 2 + O(n^2(\kappa+\delta)).$$

Similarly, $\langle \boldsymbol{b}_m, \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle$ is trivial only when $p[j] = m$ and α is the other child. By the same expansion,

$$\frac{1}{B}\sum_{\alpha\in[M]} \langle \boldsymbol{b}_m, \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle = \begin{cases} 1 + O(n(\kappa+\delta)) & p[j] = m, \\ O(n(\kappa+\delta)) & \text{o.w.} \end{cases}$$

The batch-size assumption gives $\kappa = O(n^{-1-\epsilon/4})$. Separately, $\delta = \exp(-Cn^{\epsilon/16}) = O(n^{-1-\epsilon/4})$, so $\kappa + \delta = O(n^{-1-\epsilon/4})$. Combining these bounds,

$$\begin{aligned} -\frac{1}{MB}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{b}_m, 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle &= -\frac{2c}{M^2}\mathbb{1}(p[j] = m) + O\left(M^{-3} + \frac{\kappa+\delta}{M}\right) \\ &= -\frac{2c}{M^2}\mathbb{1}(p[j] = m) + O(n^{-2-\epsilon/4}). \end{aligned}$$

Next for Equation (28), we expand

$$\begin{aligned} &\frac{1}{B}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}(-\mathbf{1}_d + c(\hat{\boldsymbol{z}}_m^{(t+1)})^2), 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle \\ =& -\frac{2c}{B}\langle \hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle + \frac{2c}{B}\langle (\hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]})^2\rangle + \frac{2c^2}{B}\langle (\hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]})^3, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle - \frac{2c^2}{B}\langle (\hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]})^4\rangle. \end{aligned}$$

For the second-order terms, we have:

$$\begin{aligned} \frac{1}{B}\langle \hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle &= \frac{1}{MB}\left(\langle \boldsymbol{x}^{(t)}_{j,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle + \sum_{\alpha\neq j,\alpha\in[M]} \langle \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{j,[t+1]}\rangle\right) \\ &= \frac{1}{MB}\left(\langle \boldsymbol{b}_j, \boldsymbol{b}_j\rangle + \sum_{\alpha\neq j}\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_j\rangle\right) + \frac{O(n)}{M}(\delta + \delta^2) \\ &= \frac{1}{M}\big(1 + O((M-1)\kappa)\big) + \frac{O(n)}{M}(\delta+\delta^2) \\ &= \frac{1}{M} + O(\kappa+\delta). \end{aligned}$$

Similarly,

$$\begin{aligned} \frac{1}{B}\langle (\hat{\boldsymbol{z}}^{(t+1)}_{m,[t+1]})^2\rangle &= \frac{1}{M^2B}\left(\sum_\alpha \langle \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{\alpha,[t+1]}\rangle + \sum_{\alpha\neq\beta}\langle \boldsymbol{x}^{(t)}_{\alpha,[t+1]}, \boldsymbol{x}^{(t)}_{\beta,[t+1]}\rangle\right) \\ &= \frac{1}{M^2B}\left(\sum_\alpha \langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\alpha\rangle + \sum_{\alpha\neq\beta}\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta\rangle\right) + \frac{O(n)}{M}\delta + \frac{O(n^2)}{M^2}\delta^2 \\ &= \frac{1}{M} + O(\kappa+\delta). \end{aligned}$$

For the fourth-order interaction terms, we discuss when $(\alpha,\beta,\gamma,\delta) \notin I_{4,m}$, i.e., $\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta\rangle$ is trivial. Again, by the bit-counting argument in Lemma 6, the number of trivial ones are at most

$O(n^2)$. By (26), each replacement contributes at most $4B\delta$; summing over M^4 tuples and dividing by M^4B leaves at most 4δ. Hence,

$$\begin{aligned}\frac{1}{B}\langle(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)})^4\rangle &= \frac{1}{M^4B}\sum_{\alpha,\beta,\gamma,\delta\in[M]}\langle \boldsymbol{x}_{\alpha,[t+1]}^{(t)}, \boldsymbol{x}_{\beta,[t+1]}^{(t)}, \boldsymbol{x}_{\gamma,[t+1]}^{(t)}, \boldsymbol{x}_{\delta,[t+1]}^{(t)}\rangle \\ &= \frac{1}{M^4B}\sum_{\alpha,\beta,\gamma,\delta\in[M]}\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_\delta\rangle + O(\delta) \\ &\le \frac{1}{M^4B}\left(\sum_{\alpha,\beta,\gamma,\delta\in I_{4,m}} O(B\kappa) + \sum_{\alpha,\beta,\gamma,\delta\notin I_{4,m}} B\right) + O(\delta) \\ &\le \frac{1}{M^4}(O(n^2) + M^4\kappa) + O(\delta) \\ &= O(n^{-2} + \kappa + \delta) = O(n^{-2} + \kappa),\end{aligned}$$

where the last equality uses $\delta = O(n^{-2})$.

Now for $\langle(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)})^3, \boldsymbol{x}_{j,[t+1]}^{(t)}\rangle$, assuming index j is contained in $(\alpha,\beta,\gamma,\delta)$. Use the absolute-value tuple bound (14) and the four-factor perturbation estimate (26). We get

$$\begin{aligned}\left|\frac{1}{B}\langle(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)})^3, \boldsymbol{x}_{j,[t+1]}^{(t)}\rangle\right| &\le \frac{1}{M^3B}\sum_{\alpha,\beta,\gamma\in[M]}|\langle \boldsymbol{b}_\alpha, \boldsymbol{b}_\beta, \boldsymbol{b}_\gamma, \boldsymbol{b}_j\rangle| + 4\delta \\ &\le \frac{1}{M^3}(O(n) + M^3\kappa) + 4\delta \\ &= O(n^{-2} + \kappa + \delta) = O(n^{-2} + \kappa).\end{aligned}$$

The leading $1/M$ terms in the two second moments cancel. Together with the fourth-order bounds, this gives

$$\begin{aligned}&\frac{1}{MB}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}(-\mathbf{1}_d + c(\hat{\boldsymbol{z}}_m^{(t+1)})^2), 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle \\ &\quad = O\left(M^{-3} + \frac{\kappa+\delta}{M}\right) = O(n^{-2-\epsilon/4}).\end{aligned}$$

For Eq. (29), note that

$$\frac{1}{B}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}|\hat{\boldsymbol{z}}_m^{(t+1)}|^4\rangle = \frac{1}{B}\langle|\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)}|^4\rangle = \frac{1}{B}\langle(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)})^4\rangle = O(n^{-2}+\kappa).$$

and since $2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}$ are contained in $[-2c, 2c]$ and $[-2, 2]$ respectively. We have

$$\frac{1}{MB}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}|\hat{\boldsymbol{z}}_m^{(t+1)}|^4, 2c\hat{\boldsymbol{z}}_m^{(t+1)}, \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle = \frac{4c\cdot O(n^{-2}+\kappa)}{M} = O(n^{-2-\epsilon/4}).$$

Finally for Eq. (30), using Cauchy-Schwarz we have

$$\begin{aligned}\frac{1}{B}\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}|\hat{\boldsymbol{z}}_m^{(t+1)}|^3\rangle = \frac{1}{B}\langle|(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)})^3|\rangle &= \frac{1}{B}\sum_{i=1}^{B}|(\hat{\boldsymbol{z}}_{m,[t+1],i}^{(t+1)})^3| \\ &\le \frac{1}{B}\left(\sum_{i=1}^{B}(\hat{\boldsymbol{z}}_{m,[t+1],i}^{(t+1)})^2\right)^{1/2}\left(\sum_{i=1}^{B}(\hat{\boldsymbol{z}}_{m,[t+1],i}^{(t+1)})^4\right)^{1/2} \\ &= \frac{1}{B}O(Bn^{-1})^{1/2}O(Bn^{-1-\epsilon/4})^{1/2} \\ &= O(n^{-1-\epsilon/8}).\end{aligned}$$

By the definition of $\boldsymbol{P}_{t+1}$, we get

$$\begin{aligned}
&\frac{1}{MB}\langle \boldsymbol{P}_{t+1}^\top(\boldsymbol{P}_{t+1}\phi(\hat{\boldsymbol{z}}_m^{(t+1)}) - \boldsymbol{b}_m), O(|\hat{\boldsymbol{z}}_m^{(t+1)}|^3), \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle \\
&= \frac{1}{MB}\langle \phi(\hat{\boldsymbol{z}}_m^{(t+1)}) - \boldsymbol{P}_{t+1}^\top \boldsymbol{b}_m, O(\boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}|\hat{\boldsymbol{z}}_m^{(t+1)}|^3), \boldsymbol{x}_j^{(t)} - \hat{\boldsymbol{z}}_m^{(t+1)}\rangle \\
&= \frac{4}{MB} O\left(\langle \boldsymbol{P}_{t+1}^\top \boldsymbol{P}_{t+1}|\hat{\boldsymbol{z}}_m^{(t+1)}|^3\rangle\right) \\
&= O(n^{-2-\epsilon/8}).
\end{aligned}$$

Combining all terms from (27) to (30), for any $j \in [M]$, we get

$$\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(t+1)}} = -\frac{2c}{M^2}\mathbb{1}(p[j] = m) + O(n^{-2-\epsilon/8})$$

Case 2: $m > n_{t+2}$**.** Under per-block gating, $g_m^{(t+1)} = 0$. The gated connection update at position m becomes

$$\boldsymbol{x}_{m,[t+2]}^{(t+1)} = \boldsymbol{x}_{m,[t+2]}^{(t)} + (1 - g_m^{(t+1)})\boldsymbol{x}_{m,[t+1]}^{(t)} + g_m^{(t+1)}\phi(\hat{\boldsymbol{z}}_m^{(t+1)}) = \boldsymbol{x}_{m,[t+2]}^{(t)} + \boldsymbol{x}_{m,[t+1]}^{(t)},$$

which does not depend on $\hat{\boldsymbol{z}}_m^{(t+1)}$ and hence not on $\boldsymbol{w}_m^{(t+1)}$. Therefore, $w_{j,m}^{(t+1)}$ is absent from the loss $\mathcal{L}^{(t+1)}$ entirely, giving

$$\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(t+1)}} = 0$$

identically. After quantization, $w_{j,m}^{(t+1)}$ remains zero.

Concentration of attention scores at $m \in [n_{t+1}+1, n_{t+2}]$**.** Similar to Lemma 7, we choose a single stage-wise learning rate

$$\eta_{t+1} := \frac{K_n n_{t+1}^2}{2c}.$$

With $M = m - 1$, put $\lambda_{t+1,m} := K_n n_{t+1}^2/M^2$. Since $n_{t+1} \le M < 2n \le 2n_{t+1}$,

$$\frac{1}{4} \le \frac{n_{t+1}^2}{M^2} \le 1, \qquad K_n/4 \le \lambda_{t+1,m} \le K_n.$$

The layer is still at zero initialization, so

$$\begin{aligned}
w_{j,m}^{(t+1)}(t+1) &= q\Big(-\eta_{t+1}\widetilde{\nabla}_{w_{j,m}^{(t+1)}}\mathcal{L}^{(t+1)}\Big) \\
&= q(\lambda_{t+1,m}\mathbb{1}(p[j] = m) + \Delta_{j,m}),
\end{aligned} \tag{31}$$

where $\Delta_{j,m} := \eta_{t+1}R_{j,m}$ and $R_{j,m}$ includes the coordinatewise oracle error in the negative gradient. The termwise calculation above gives, uniformly over $j < m$,

$$|R_{j,m}| \le C_\nabla n^{-2-\epsilon/8}, \qquad |\Delta_{j,m}| \le C'_\nabla n^{-\epsilon/16} < 1/4.$$

The scaled remainder obeys the calculation in (16), with n^2 replaced by $n_{t+1}^2 \le 4n^2$.

Non-child indices. If $p[j] \ne m$, then $q(\Delta_{j,m}) = 0$ because $|\Delta_{j,m}| < 1/2$.

Children indices. Both real-valued child updates lie within $1/4$ of $\lambda_{t+1,m}$. Applying the nearest-integer calculation leading to (17), with λ_m replaced by $\lambda_{t+1,m}$, gives

$$\min_{i\in\{1,2\}} w_{c_i[m],m}^{(t+1)}(t+1) \ge K_n/4 - 3/4 \ge K_n/5,$$

$$\left|w_{c_1[m],m}^{(t+1)}(t+1) - w_{c_2[m],m}^{(t+1)}(t+1)\right| \le 1.$$

Softmax concentration. Applying (18) to the $M-2$ zero non-child logits and the two child logits just bounded gives

$$\mu_m^{(t+1)} := \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}_m^{(t+1)}) \le \frac{M-2}{2e^{K_n/5}+M-2} \le 2ne^{-K_n/5} \le \exp(-Cn^{\epsilon/16}).$$

The child-probability calculation in (19) then gives

$$\frac{1-\mu_m^{(t+1)}}{1+e} \le \sigma_{c_i[m]}(\boldsymbol{w}_m^{(t+1)}) \le \frac{e}{1+e}, \qquad i \in \{1,2\}.$$

Evaluating the forward pass. Fix any input-bit batch $\boldsymbol{b}_1, \dots, \boldsymbol{b}_n \in \{\pm 1\}^B$ and any initial internal-token contents $\boldsymbol{x}_{j,[1]}^{(0)} \in [-1,1]^B$ for $j > n$, with all higher residual blocks initialized to zero. Let $\boldsymbol{b}_m$ denote the ground-truth parity vector determined by these input bits. Holding this input configuration and the network weights fixed, define the cumulative approximation error through layer t by

$$\epsilon_t := \max_{1\le s\le t} \max_{m\in[n_s+1,n_{s+1}]} \|\boldsymbol{x}_{m,[s+1]}^{(s)} - \boldsymbol{b}_m\|_\infty, \quad \epsilon_0 := 0.$$

By construction, ϵ_t is non-decreasing in t and bounds the error at every CoT position produced by layers $1,\dots,t$. In particular, for any $m \in [n_{t+1}+1, n_{t+2}]$, the two children were produced by some layer $s \le t$ and satisfy

$$\|\boldsymbol{x}_{c_i[m],[t+1]}^{(t)} - \boldsymbol{b}_{c_i[m]}\|_\infty \le \epsilon_t, \qquad i \in \{1,2\}.$$

Define the child-renormalized average of the exact parity vectors by

$$\theta_m := \frac{e^{w_{c_1[m],m}^{(t+1)}}}{e^{w_{c_1[m],m}^{(t+1)}} + e^{w_{c_2[m],m}^{(t+1)}}}, \qquad \boldsymbol{s}_m := \theta_m \boldsymbol{b}_{c_1[m]} + (1-\theta_m)\boldsymbol{b}_{c_2[m]}.$$

The child logits are integers at the algorithm iterate and differ by at most one. Thus (20) applies, giving $\phi(\boldsymbol{s}_m) = \boldsymbol{b}_m$ and $\phi'(\boldsymbol{s}_m) = \boldsymbol{0}_B$. All values are bounded by (7). Write the child-renormalized average of the actual hidden values as

$$\boldsymbol{h}_m := \theta_m \boldsymbol{x}_{c_1[m],[t+1]}^{(t)} + (1-\theta_m)\boldsymbol{x}_{c_2[m],[t+1]}^{(t)}.$$

The two child errors give

$$\begin{aligned}\|\boldsymbol{h}_m - \boldsymbol{s}_m\|_\infty &\le \theta_m \|\boldsymbol{x}_{c_1[m],[t+1]}^{(t)} - \boldsymbol{b}_{c_1[m]}\|_\infty \\ &\quad + (1-\theta_m)\|\boldsymbol{x}_{c_2[m],[t+1]}^{(t)} - \boldsymbol{b}_{c_2[m]}\|_\infty \le \epsilon_t.\end{aligned}$$

As in (21),

$$\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)} - \boldsymbol{h}_m = \sum_{j<m,\, p[j]\neq m} \sigma_j(\boldsymbol{w}_m^{(t+1)})\boldsymbol{x}_{j,[t+1]}^{(t)} - \mu_m^{(t+1)}\boldsymbol{h}_m.$$

Its norm is at most $2\mu_m^{(t+1)}$, because $\|\boldsymbol{h}_m\|_\infty \le 1$ and the non-child attention mass is $\mu_m^{(t+1)}$. The triangle inequality now gives

$$\begin{aligned}\|\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)} - \boldsymbol{s}_m\|_\infty &\le \|\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)} - \boldsymbol{h}_m\|_\infty + \|\boldsymbol{h}_m - \boldsymbol{s}_m\|_\infty \\ &\le 2\mu_m^{(t+1)} + \epsilon_t \le 2\exp(-Cn^{\epsilon/16}) + \epsilon_t. \end{aligned} \tag{32}$$

The coordinatewise Taylor bound (22) at $\boldsymbol{s}_m$ gives

$$\begin{aligned}\|\boldsymbol{x}_{m,[t+2]}^{(t+1)} - \boldsymbol{b}_m\|_\infty &= \|\phi(\hat{\boldsymbol{z}}_{m,[t+1]}^{(t+1)}) - \phi(\boldsymbol{s}_m)\|_\infty \\ &\le C_2\big(2\exp(-Cn^{\epsilon/16}) + \epsilon_t\big)^2.\end{aligned}$$

Combined with the cumulative max, this gives the recursion

$$\epsilon_{t+1} \le \max\Big\{\epsilon_t,\ C_2\big(2\exp(-Cn^{\epsilon/16}) + \epsilon_t\big)^2\Big\}.$$

We inductively show that $\epsilon_t \le \delta := \exp(-Cn^{\epsilon/16})$ for all t. The base case $\epsilon_1 \le \delta$ holds by Lemma 7. Assume $\epsilon_t \le \delta$. The first argument of the max is at most δ by hypothesis, and the second is bounded by

$$C_2(2\delta + \delta)^2 = 9C_2\delta^2.$$

To conclude $\epsilon_{t+1} \le \delta$, it suffices that $9C_2\delta \le 1$, which holds for all sufficiently large n. Thus $\epsilon_t \le \delta$ for all $t \le L = \log_2 k$. The input configuration was arbitrary and the constants do not depend on it, so this bound holds uniformly over the stated input bits and initial internal-token contents. In particular, choosing the all-zero internal-token test configuration and position $m = T$ gives

$$\|\boldsymbol{x}^{(L)}_{T,[L+1]} - \boldsymbol{b}_T\|_\infty = \|\hat{f}(\boldsymbol{D}_{\text{test}}) - \boldsymbol{y}_{\text{test}}\|_\infty \le \exp(-Cn^{\epsilon/16}).$$

□

We are now ready to combine the lemmas of this section into a proof of Theorem 1.

Proof of Theorem 1. We prove by induction on $t \in \{1, \dots, L\}$ that after stage t, layers $1, \dots, t$ are well-trained (Definition 4) and $\boldsymbol{W}^{(\ell)}(t) = 0$ for all $\ell \ge t + 1$.

Base case ($t = 1$). At initialization, $\boldsymbol{W}^{(\ell)}(0) = 0$ for all $\ell \in [L]$ (Algorithm 1). The stage-1 update modifies only $\boldsymbol{W}^{(1)}$, leaving $\boldsymbol{W}^{(\ell)}(1) = 0$ for all $\ell \ge 2$. The readout at stage t extracts block $[t + 1]$, which is written using only layers $1, \dots, t$; its exact gradient with respect to later layers is therefore zero. A column whose query has zero gate at its layer cannot affect that layer's update, and a masked key cannot affect its softmax, so these coordinates also have zero exact gradient. For any such initially zero logit, the coordinatewise oracle error gives

$$\left|\eta_t \widetilde{\nabla}_w \mathcal{L}^{(t)}\right| \le \eta_t \delta_n = O\Big(K_n n_t^2 n^{-2-\epsilon/8}\Big) = O(n^{-\epsilon/16}) < 1/2,$$

using the scaling calculation in (16) and $n_t < 2n$. Thus its update is $q(0 - \eta_t \widetilde{\nabla}_w \mathcal{L}^{(t)}) = 0$ at every stage. By Lemma 7, with probability $1 - \exp(-n^{\epsilon/2})$ over the stage-1 batch, layer 1 is well-trained.

Inductive step ($t \to t+1$). Assume that after stage t, layers $1, \dots, t$ are well-trained and $\boldsymbol{W}^{(\ell)}(t) = 0$ for all $\ell \ge t + 1$. We show the same holds at stage $t + 1$.

The inductive hypothesis verifies the assumption of Lemma 10. Applying it, for every $\ell' \le t$, $m \in [n_{\ell'} + 1, n_{\ell'+1}]$, and $j < m$,

$$\left|\frac{\partial \mathcal{L}^{(t+1)}}{\partial w^{(\ell')}_{j,m}}\right| \le O\big(\exp(-Cn^{\epsilon/16}/2)\big),$$

and the quantized stage-$(t + 1)$ update leaves $w^{(\ell')}_{j,m}(t + 1) = w^{(\ell')}_{j,m}(t)$. Hence layers $1, \dots, t$ remain well-trained after stage $t + 1$. The inductive hypothesis also verifies the assumption of Lemma 9. Applying it, with probability $1 - \exp(-n^{\epsilon/2})$ over the fresh stage-$(t + 1)$ batch, layer $t + 1$ is well-trained. The stage-$(t + 1)$ update modifies only $\boldsymbol{W}^{(t+1)}$, so $\boldsymbol{W}^{(\ell)}(t + 1) = \boldsymbol{W}^{(\ell)}(t) = 0$ for all $\ell \ge t + 2$. This completes the inductive step.

Union bound and conclusion. Each stage's well-trainedness conclusion holds with probability $1 - \exp(-n^{\epsilon/2})$ over its fresh batch, and the $L = \log_2 k$ batches are independent. By a union bound, with probability at least

$$1 - L\exp(-n^{\epsilon/2}) = 1 - \exp(-\Omega(n^{\epsilon/2}))$$

(using $L = O(\log n)$), every layer $\ell \in [L]$ is well-trained after stage L. By Definition 4 with the all-zero internal-token test configuration, applied at layer L and position $m = T$ (which lies in $[n_L + 1, n_{L+1}]$ since $n_{L+1} = T$),

$$\|\boldsymbol{x}^{(L)}_{T,[L+1]} - \boldsymbol{b}_T\|_\infty \le \exp(-Cn^{\epsilon/16}).$$

By Lemma 2 applied at $\beta = T$, $\ell = L$, the test-time output at position T satisfies

$$f^L_{\hat{\boldsymbol{\theta}}}(\boldsymbol{D}_{\text{test}})_T = \Psi_L^\top \mathcal{T}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{D}_{\text{test}})_T = \boldsymbol{x}^{(L)}_{T,[L+1]},$$

and the target is $\boldsymbol{y}_{\text{test}} = \boldsymbol{b}_T$. Therefore

$$\|f_{\hat{\boldsymbol{\theta}}}^L(\boldsymbol{D}_{\text{test}})_T - \boldsymbol{y}_{\text{test}}\|_\infty \le \exp(-Cn^{\epsilon/16}) = \exp(-\Omega(n^{\epsilon/16})),$$

which establishes the theorem. □

C.6 Gradient Upper Bound for Trained Layers

Lemma 10 (Frozen gradient)**.** *Fix any $t \in \{1, \dots, L-1\}$. At the stage-t iterate of Algorithm 1, suppose layers $1, \dots, t$ are well-trained (Definition 4) and $\boldsymbol{W}^{(\ell)}(t) = 0$ for all $\ell \ge t+1$. Then during one stage-$(t+1)$ gradient step, for any $\ell' \le t$, any $m \in [n_{\ell'}+1, n_{\ell'+1}]$, and any $j < m$,*

$$\left|\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(\ell')}}\right| \le O\big(\exp(-Cn^{\epsilon/16}/2)\big).$$

Consequently, with the stage-wise learning rate $\eta_{t+1} = \Theta(K_n n_{t+1}^2)$ (where $K_n := \lceil n^{\epsilon/16} \rceil$), the weights in all previously trained layers remain unchanged after quantization:

$$w_{j,m}^{(\ell')}(t+1) = w_{j,m}^{(\ell')}(t) \quad \textit{for all } \ell' \le t,\ m \in [n_{\ell'}+1, n_{\ell'+1}],\ j < m.$$

Proof. Throughout, fix $\ell' \le t$ and a parameter $w_{j,m}^{(\ell')}$ with $n_{\ell'}+1 \le m \le n_{\ell'+1}$ and $1 \le j < m$. Let $L_\phi := \|\phi'\|_\infty$.

Step 1: Vanishing of ϕ' at well-trained positions. For each $\ell'' \le t$ and $\alpha \in [n_{\ell''}+1, n_{\ell''+1}]$, normalize the attention over the children:

$$\theta_\alpha^{(\ell'')} := \frac{e^{w_{c_1[\alpha],\alpha}^{(\ell'')}}}{e^{w_{c_1[\alpha],\alpha}^{(\ell'')}} + e^{w_{c_2[\alpha],\alpha}^{(\ell'')}}},$$
$$\boldsymbol{s}_\alpha^{(\ell'')} := \theta_\alpha^{(\ell'')} \boldsymbol{b}_{c_1[\alpha]} + (1 - \theta_\alpha^{(\ell'')}) \boldsymbol{b}_{c_2[\alpha]}.$$

The preceding updates in Algorithm 1 quantize the child logits to integers. Together with the gap bound in Definition 4, their difference belongs to $\{-1, 0, 1\}$. The calculation in (20) therefore places the coordinates of $\boldsymbol{s}_\alpha^{(\ell'')}$ in $\mathcal{Z} = \{-1, -r, 0, r, 1\}$. The uniform forward bound of Definition 4 applies to the fresh stage-$(t+1)$ batch. By Lemma 2, each child is either an exact input sign or inherits its already-computed value within error $\delta = \exp(-Cn^{\epsilon/16})$. All other causal values are bounded by (7). Thus the leakage and child-error decomposition in (32), with child error at most δ, gives

$$\|\hat{\boldsymbol{z}}_{\alpha,[\ell'']}^{(\ell'')} - \boldsymbol{s}_\alpha^{(\ell'')}\|_\infty \le 2\mu_\alpha^{(\ell'')} + \delta \le 3\exp(-Cn^{\epsilon/16}). \tag{33}$$

By Definition 2, ϕ' vanishes at every coordinate of $\boldsymbol{s}_\alpha^{(\ell'')}$. With $C_\phi := \sup_{|v|\le 1} |\phi''(v)|$, the mean value theorem gives

$$|\phi'(u)| \le C_\phi |u - z_i^\star|, \qquad u \in [-1, 1], \quad z_i^\star \in \mathcal{Z}. \tag{34}$$

Applying this bound with $z_i^\star = (\boldsymbol{s}_\alpha^{(\ell'')})_i$ yields

$$\begin{aligned}\|\phi'(\hat{\boldsymbol{z}}_{\alpha,[\ell'']}^{(\ell'')})\|_\infty &\le C_\phi \|\hat{\boldsymbol{z}}_{\alpha,[\ell'']}^{(\ell'')} - \boldsymbol{s}_\alpha^{(\ell'')}\|_\infty \\ &\le O(\exp(-Cn^{\epsilon/16})).\end{aligned}$$

Step 2: Per-node sensitivity under the well-trained hypothesis. Recall the per-node sensitivity defined in Lemma 3:

$$\xi_{j,m,\ell'}^{(\ell)} := \max_{\beta\,:\,h[\beta] \le \ell+1} \left\| \frac{\partial \boldsymbol{x}_{\beta,[\ell+1]}^{(\ell)}}{\partial w_{j,m}^{(\ell')}} \right\|_\infty.$$

We claim that for all $\ell' \le \ell \le t$,

$$\xi^{(\ell)}_{j,m,\ell'} \le O(\exp(-Cn^{\epsilon/16})).$$

Base case $\ell = \ell'$. By Corollary 1, only β with active layer equal to ℓ', i.e., $\nu := h[\beta] = \ell' + 1$, contribute. For such β,

$$\frac{\partial \boldsymbol{x}^{(\ell')}_{\beta,[\ell'+1]}}{\partial w^{(\ell')}_{j,m}} = \phi'(\hat{\boldsymbol{z}}^{(\ell')}_{\beta,[\ell']}) \odot \frac{\partial \hat{\boldsymbol{z}}^{(\ell')}_{\beta,[\ell']}}{\partial w^{(\ell')}_{j,m}},$$

and the parameter $w^{(\ell')}_{j,m}$ enters only through the softmax at query position $\beta = m$, giving

$$\frac{\partial \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']}}{\partial w^{(\ell')}_{j,m}} = \sigma_j(\boldsymbol{w}^{(\ell')}_m)\big(\boldsymbol{x}^{(\ell'-1)}_{j,[\ell']} - \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']}\big).$$

Since $\boldsymbol{x}^{(\ell'-1)}_{j,[\ell']}, \hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']} \in [-1,1]^B$, their difference lies in $[-2,2]^B$. Combining with Step 1,

$$\xi^{(\ell')}_{j,m,\ell'} \le 2\,\|\phi'(\hat{\boldsymbol{z}}^{(\ell')}_{m,[\ell']})\|_\infty \cdot \sigma_j(\boldsymbol{w}^{(\ell')}_m) \le 6C_\phi\delta.$$

Inductive step $\ell \Rightarrow \ell+1$ *(*$\ell + 1 \le t$*).* Take β with $h[\beta] \le \ell + 2$.

Sub-case A: $h[\beta] \le \ell + 1$. By Lemma 2, $\boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]} = \boldsymbol{x}^{(\ell)}_{\beta,[\ell+1]}$, so the derivative is bounded by $\xi^{(\ell)}_{j,m,\ell'}$.

Sub-case B: $h[\beta] = \ell + 2$. Corollary 1 gives

$$\frac{\partial \boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}}{\partial w^{(\ell')}_{j,m}} = \phi'(\hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}) \odot \frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}}.$$

Since $\ell+1 \le t$, Step 1 yields $\|\phi'(\hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]})\|_\infty \le O(\exp(-Cn^{\epsilon/16}))$. Differentiation passes through the softmax (since $\ell' \le \ell < \ell + 1$):

$$\frac{\partial \hat{\boldsymbol{z}}^{(\ell+1)}_{\beta,[\ell+1]}}{\partial w^{(\ell')}_{j,m}} = \sum_{\gamma<\beta} \sigma_\gamma(\boldsymbol{w}^{(\ell+1)}_\beta)\, \frac{\partial \boldsymbol{x}^{(\ell)}_{\gamma,[\ell+1]}}{\partial w^{(\ell')}_{j,m}},$$

The restricted-prefix calculation in (8) removes the zero derivatives of unwritten causal peers and bounds the remaining weighted sum by $\xi^{(\ell)}_{j,m,\ell'}$. Hence

$$\left\| \frac{\partial \boldsymbol{x}^{(\ell+1)}_{\beta,[\ell+2]}}{\partial w^{(\ell')}_{j,m}} \right\|_\infty \le 3C_\phi\delta \cdot \xi^{(\ell)}_{j,m,\ell'}.$$

Combining sub-cases.

$$\xi^{(\ell+1)}_{j,m,\ell'} \le \xi^{(\ell)}_{j,m,\ell'} \le O(\exp(-Cn^{\epsilon/16})),$$

For sufficiently large n, $3C_\phi\delta \le 1$, so the newly written sub-case cannot increase the sensitivity. Starting from the base bound above, induction gives $\xi^{(\ell)}_{j,m,\ell'} \le 6C_\phi\delta$ for every $\ell \in \{\ell', \dots, t\}$.

Step 3: Bounding the layer-$(t+1)$ output gradient. For any $\alpha \in [n_{t+1}+1, T]$, the softmax weights at layer $t+1$ are independent of $w^{(\ell')}_{j,m}$ as layer $t+1$ has not yet been trained. The zero-peer derivative calculation used in (8), now at $\ell = t$, applies to every key with $h[\gamma] > t+1$. By chain rule,

$$\frac{\partial \phi(\hat{\boldsymbol{z}}^{(t+1)}_{\alpha,[t+1]})}{\partial w^{(\ell')}_{j,m}} = \phi'(\hat{\boldsymbol{z}}^{(t+1)}_{\alpha,[t+1]}) \odot \sum_{\gamma<\alpha} \sigma_\gamma(\boldsymbol{w}^{(t+1)}_\alpha) \cdot \frac{\partial \boldsymbol{x}^{(t)}_{\gamma,[t+1]}}{\partial w^{(\ell')}_{j,m}}.$$

Bounding by $\|\phi'\|_\infty \le L_\phi$, summing softmax weights to 1, and using Step 2's bound $\xi^{(t)}_{j,m,\ell'} \le O(\exp(-Cn^{\epsilon/16}))$,

$$\left\| \frac{\partial \phi(\hat{\boldsymbol{z}}^{(t+1)}_{\alpha,[t+1]})}{\partial w^{(\ell')}_{j,m}} \right\|_\infty \le L_\phi \cdot \xi^{(t)}_{j,m,\ell'} \le 6L_\phi C_\phi\delta.$$

Step 4: Pulling back through the loss. The stage-$(t+1)$ loss reads

$$\mathcal{L}^{(t+1)}(\boldsymbol{\theta}) = \frac{1}{2B} \sum_{\alpha=n_{t+1}+1}^{T} \|\boldsymbol{P}_{t+1}\left(g_\alpha^{(t+1)}\phi(\hat{\boldsymbol{z}}_\alpha^{(t+1)}) + (1-g_\alpha^{(t+1)})\boldsymbol{x}_\alpha^{(t)}\right) - \boldsymbol{b}_\alpha\|^2.$$

Splitting the sum by the gating rule Eq. (4), $g_\alpha^{(t+1)} = 1$ for $\alpha \in [n_{t+1}+1, n_{t+2}]$ and $g_\alpha^{(t+1)} = 0$ for $\alpha > n_{t+2}$ and differentiating,

$$\begin{aligned}
&\left|\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(\ell')}}\right| \\
&= \frac{1}{B}\Bigg|\sum_{\alpha=n_{t+1}+1}^{n_{t+2}} \left(\boldsymbol{P}_{t+1}\phi(\hat{\boldsymbol{z}}_\alpha^{(t+1)}) - \boldsymbol{b}_\alpha\right)^\top \frac{\partial \boldsymbol{P}_{t+1}\phi(\hat{\boldsymbol{z}}_\alpha^{(t+1)})}{\partial w_{j,m}^{(\ell')}} \\
&\quad + \sum_{\alpha=n_{t+2}+1}^{T} \left(\boldsymbol{P}_{t+1}\boldsymbol{x}_\alpha^{(t)} - \boldsymbol{b}_\alpha\right)^\top \frac{\partial \boldsymbol{P}_{t+1}\boldsymbol{x}_\alpha^{(t)}}{\partial w_{j,m}^{(\ell')}}\Bigg| \\
&\overset{(a)}{\leq} \sum_{\alpha=n_{t+1}+1}^{n_{t+2}} \frac{1}{B}\|\boldsymbol{P}_{t+1}\phi(\hat{\boldsymbol{z}}_\alpha^{(t+1)}) - \boldsymbol{b}_\alpha\|_1 \cdot \left\|\frac{\partial \phi(\hat{\boldsymbol{z}}_{\alpha,[t+1]}^{(t+1)})}{\partial w_{j,m}^{(\ell')}}\right\|_\infty \\
&\quad + \sum_{\alpha=n_{t+2}+1}^{T} \frac{1}{B}\|\boldsymbol{P}_{t+1}\boldsymbol{x}_\alpha^{(t)} - \boldsymbol{b}_\alpha\|_1 \cdot \left\|\frac{\partial \boldsymbol{x}_{\alpha,[t+1]}^{(t)}}{\partial w_{j,m}^{(\ell')}}\right\|_\infty \\
&\overset{(b)}{\leq} 2\sum_{\alpha=n_{t+1}+1}^{n_{t+2}} \left\|\frac{\partial \phi(\hat{\boldsymbol{z}}_{\alpha,[t+1]}^{(t+1)})}{\partial w_{j,m}^{(\ell')}}\right\|_\infty + 2\sum_{\alpha=n_{t+2}+1}^{T} \left\|\frac{\partial \boldsymbol{x}_{\alpha,[t+1]}^{(t)}}{\partial w_{j,m}^{(\ell')}}\right\|_\infty \\
&\overset{(c)}{\leq} 12TL_\phi C_\phi \delta,
\end{aligned}$$

where (a) applies Hölder's inequality and uses that $\boldsymbol{P}_{t+1}$ extracts block $[t+1]$, (b) uses $\boldsymbol{P}_{t+1}\phi(\hat{\boldsymbol{z}}_\alpha^{(t+1)})$, $\boldsymbol{P}_{t+1}\boldsymbol{x}_\alpha^{(t)}, \boldsymbol{b}_\alpha \in [-1,1]^B$, and (c) bounds the first sum by Step 3. The second sum is zero: if $\alpha > n_{t+2}$, then $h[\alpha] > t+2$ and Corollary 1 gives $\partial \boldsymbol{x}_{\alpha,[t+1]}^{(t)}/\partial w_{j,m}^{(\ell')} = 0$. The first sum has at most T terms, each at most $2 \cdot 6L_\phi C_\phi \delta$. Since $T < 2n$ and L_ϕ, C_ϕ are fixed constants, for sufficiently large n we have $\log(12TL_\phi C_\phi) \leq Cn^{\epsilon/16}/2$. Therefore

$$12TL_\phi C_\phi \delta = \exp\Big(\log(12TL_\phi C_\phi) - Cn^{\epsilon/16}\Big) \leq \exp(-Cn^{\epsilon/16}/2).$$

This explicitly accounts for the polynomial number of loss terms by reducing the exponential constant.

Step 5: Quantization freezes the weight. With the stage-wise learning rate $\eta_{t+1} = \Theta(K_n n_{t+1}^2)$ where $K_n = \lceil n^{\epsilon/16} \rceil$, the coordinatewise oracle error $\delta_n = O(n^{-2-\epsilon/8})$ gives

$$\begin{aligned}
\left|\eta_{t+1}\widetilde{\nabla}_{w_{j,m}^{(\ell')}}\mathcal{L}^{(t+1)}\right| &\leq \eta_{t+1}\left(\left|\frac{\partial \mathcal{L}^{(t+1)}}{\partial w_{j,m}^{(\ell')}}\right| + \delta_n\right) \\
&= O\Big(n^{2+\epsilon/16}\exp(-Cn^{\epsilon/16}/2)\Big) + O\Big(n^{2+\epsilon/16}n^{-2-\epsilon/8}\Big) \\
&= O(n^{-\epsilon/16}) < 1/2.
\end{aligned}$$

Here $n_{t+1} < 2n$ and $K_n \leq 2n^{\epsilon/16}$ bound the learning rate, and the exponential term is smaller than $n^{-\epsilon/16}$ for sufficiently large n. The current logit $w_{j,m}^{(\ell')}(t)$ is an integer because it is an algorithm iterate. Its proposed change is strictly below $1/2$, so

$$q\Big(w_{j,m}^{(\ell')}(t) - \eta_{t+1}\widetilde{\nabla}_{w_{j,m}^{(\ell')}}\mathcal{L}^{(t+1)}\Big) = w_{j,m}^{(\ell')}(t).$$

□

D FINAL-LABEL APPROXIMATE-GRADIENT LOWER BOUND

For the lower-bound comparison in this appendix, assume $\delta n \le k \le (1-\delta)n$ for a fixed constant $\delta \in (0, 1/2)$ independent of n.

The learner receives B i.i.d. samples $\{(\boldsymbol{b}^i, y^i)\}_{i=1}^B$ from the same secret set S. For a differentiable predictor $f_\theta : \{\pm 1\}^n \to \mathbb{R}$, define the final-label loss

$$\mathcal{L}_B(\theta) := \frac{1}{2B} \sum_{i=1}^{B} \big(y^i - f_\theta(\boldsymbol{b}^i)\big)^2.$$

For the random-task lower bound, draw $S \sim \mathsf{Unif}(\mathcal{S})$ and write $\mathcal{E}(\theta) := \mathbb{E}_S[\mathcal{E}_S(\theta)]$, with the learned parameter understood to depend on the training data for S. An ε-approximate gradient oracle for $\mathcal{L}_B$ is a possibly randomized map $\tilde{\nabla}\mathcal{L}_B$ satisfying $\|\tilde{\nabla}\mathcal{L}_B(\theta) - \nabla\mathcal{L}_B(\theta)\|_2 \le \varepsilon$ for all queried θ.

Proposition 1 (Kim & Suzuki (2024), Theorem 2)**.** *Fix constants $\nu_1, \nu_2, \nu_3, \nu_4 \ge 0$ independent of n and define $\nu := 4\nu_1 + 4\nu_2 + 2\nu_3 + 2\nu_4 + 1$. Suppose $B = \Omega(n^\nu)$ and $\|\nabla f_\theta\| = O(n^{\nu_1})$. Then there exists an $O(n^{-\nu_2})$-accurate gradient oracle $\tilde{\nabla}$ such that, with probability at least $1 - \exp(-\Omega(n))$ over the random sampling, the output $\theta(\mathcal{A})$ of any iterative algorithm $\mathcal{A}$ making at most $O(n^{\nu_3})$ queries to $\tilde{\nabla}\mathcal{L}_B$ satisfies $\mathcal{E}(\theta(\mathcal{A})) \ge 1 - O(n^{-\nu_4})$.*